Unsupervised learning from textual data with neural text representations

Mira Ait Saada

Université Paris Cité, Centre Borelli (UMR 9010)

Caisse des Dépôts et Consignations, Datalab, Fabrique Digitale

Ecole Doctorale Informatique, Télécommunications et Electronique,
EDITE de Paris (ED130)

Thèse

Unsupervised Learning from Textual Data with Neural Text Representations

par

Mira Ait Saada

Thèse présentée en vue de l'obtention du grade de

Docteur en Informatique

Spécialité : Science des données

Soutenue le 18 avril 2023

Membres du jury :

Mohamed Nadif	Université Paris Cité	Directeur de thèse
Jean-Charles Lamirel	Loria, Université de Lorraine	Rapporteur
Mohamed Quafafou	LIS, Université d'Aix-Marseille	Rapporteur
Véronique Cariou	StatSC, Oniris	Examinatrice
Sophie Rosset	LISN, Université Paris Saclay	Examinatrice
Abdou Fall	Caisse des Dépôts et Consignations	Encadrant industriel

To my parents.

Remerciements

Je tiens tout d'abord à exprimer toute ma gratitude à mon directeur de thèse, Mohamed Nadif, dont les conseils et l'expérience m'ont tant appris et dont les encouragements et la confiance m'ont permis de mener à terme cette thèse. Je remercie également sincèrement Abdou Fall et Philippe Caila pour cette formidable opportunité, travailler auprès d'eux a été une chance, tant humainement que professionnellement. Merci également à toutes les personnes avec qui j'ai eu le plaisir de collaborer au niveau académique durant ces trois années, je pense notamment à François Role qui m'a beaucoup apporté et qui m'a appris à mettre en valeur mes travaux, ainsi qu'à Rafika Boutalbi, que je remercie pour les nombreux échanges constructifs que l'ont a eus. Un grand merci également aux rapporteurs M. Jean-Charles Lamirel et M. Mohamed Quafafou ainsi qu'aux examinatrices Mme Véronique Cariou et Mme Sophie Rosset de me faire l'honneur d'évaluer mon travail.

Du coté de la Fabrique, la liste des personnes à remercier est longue; d'abord merci à mes coéquipiers, actuels et passés, du Datalab : Karine Hoorens que je remercie d'être aussi bon public et de rire à mes blagues les plus nulles, Willie Béatrix Drouhet que je remercie pour ses questions pertinentes et que je ne remercie pas de me laisser (même s'il le nie) marquer des points au ping-pong, Stéphane Prioux que je remercie en particulier de m'avoir sauvé la mise en août dernier (je ne l'oublie pas !) et Gautier Solard que je remercie d'avoir été un bon maître d'apprentissage ainsi que d'être allé chercher le colis à l'accueil quand je le lui ai si gentiment demandé. Merci à tous les quatre pour votre esprit d'équipe, votre soutien et votre bonne humeur. Merci également aux membres de la code factory : Joel Desplanches, Hugo Avril, Serkan Azap, Axel Zavier, Chakib Fettal, Salimata Moustapha-Alifei, Marc Gnanou, Laurent Quastana, Abou-Oumar Diallo et j'en passe, pour leur aide précieuse et pour les bons moments passés à la Fabrique. Merci aussi à Rosa Gerber et Dalila Bessaa pour leur disponibilité et leur aide inestimable. Je remercie également mes collègues de la DFC, qui m'ont aidée à développer la partie applicative de cette thèse et notamment Laurent Durain, Philippe Treilhou et Olivier Bertrand.

Cette thèse a été financée par la CDC et l'ANRT, que je remercie vivement d'avoir rendu possible ce travail. Enfin, merci à toutes les personnes, permanentes ou de passage, avec qui j'ai pu collaborer durant les trois dernières années.

Titre : Apprentissage non-supervisé sur des données textuelles à partir de représentations neuronales de texte

Résumé : Les techniques de plongement de mots sont des techniques qui ont pour but de calculer une représentation vectorielle d'un mot, en capturant son sens sémantique. Ces représentations sont destinées à alimenter toutes sortes d'algorithmes d'apprentissage automatique. Toutefois, force est de constater que les techniques de plongement, notamment les modèles Transformeur, n'ont été que très peu exploités dans le monde non-supervisé. Dans cette thèse, nous nous intéressons exclusivement à des tâches d'apprentissage non-supervisé comme l'apprentissage automatique et la détection d'anomalies. Dans un premier temps, nous tentons de décortiquer ces modèles multi-couches en gardant en tête le contexte non-supervisé. Dans un second temps, nous proposons plusieurs méthodologies permettant d'exploiter au mieux les modèles Transformeur, obtenant des résultats satisfaisant sur des jeux de données réels.

Mots clés : Apprentissage non-supervisé, Plongements de mots, Modèles Transformeur, Classification automatique, Détection d'anomalies.

Title: Unsupervised Learning from Textual Data with Neural Text Representations

Abstract: Word embeddings are techniques that aim at computing a vector representation of words, capturing their semantic meaning. These representations are intended to be fed into various kinds of machine learning algorithms. However, we observed that embedding techniques, notably Transformer models, have been overlooked in the unsupervised world of Natural Language Processing. In this thesis, we focus exclusively on unsupervised learning tasks such as clustering and anomaly detection. First, we attempt to dissect these multi-layer black-box models while keeping in mind the unsupervised context. Second, we propose several novel methodologies to make the most of Transformer models, obtaining satisfactory results on real datasets.

Keywords: Unsupervised learning, Word embeddings, Transformer models, Clustering, Anomaly detection.

Résumé substantiel

L'ère du numérique induit indubitablement des volumes colossaux de données dont la majeure partie est non-structurée, ce qui signifie qu'elles ne sont pas organisées sous une forme déterminée comme par exemple un tableau à valeurs numériques. Les images et les documents font partie de ces données et nécessitent des traitements particuliers afin d'en tirer de la valeur. Une difficulté supplémentaire se présente vis-à-vis des données textuelles, car celles-ci, dans leur forme brute, ne contiennent pas de valeurs numériques, contrairement aux images, par exemple. Dans ce cas, la question que l'on pourrait se poser est comment un algorithme peut comprendre que «bénéfique» et «utile» ont un sens similaire, alors qu'«utile» et «futile» ont un sens opposé ? Pour répondre à cela, des techniques ont été mises en place afin de transformer automatiquement du texte en données numériques qu'un algorithme d'apprentissage automatique peut traiter. Ces méthodes sont appelées *plongements de mots* et permettent de transformer des mots ou des séquences de texte en vecteurs à valeurs réelles et de même taille. Ces plongements capturent efficacement la sémantique de chaque mot et permettent, par exemple, de savoir dans quelle mesure deux phrases veulent dire la même chose.

Au cours de la dernière décennie, le domaine des plongements de mots a connu un engouement sans précédent notamment avec l'avènement des modèles Transformeurs, avec une multitude de méthodes ayant vu le jour pour répondre à toutes sortes de problèmes de traitement automatique de la langue comme la reconnaissance d'entités nommées, l'analyse de sentiment ainsi que les systèmes de question-réponse. Toutes ces tâches sont dites supervisées et nécessitent une labellisation manuelle afin d'entraîner les modèles apprenants. Dans cette thèse en revanche, nous nous plaçons dans un contexte non-supervisé où nous supposons ne disposer d'aucune information experte vis-à-vis des données. Les tâches non-supervisées constituent un enjeu majeur dans l'industrie, au vu de la quantité de données disponible et du coût de la labellisation qui nécessite un investissement humain considérable. L'apprentissage non-supervisé permet de créer de la valeur à partir de larges volumes de données et a un fort intérêt exploratoire et d'aide à la décision. Parmi les tâches non-supervisées les plus importantes, on retrouve la classification automatique (ou

clustering), la détection d'anomalies et la visualisation de données. C'est autour de ces thématiques que s'articulent les travaux de cette thèse.

Tout d'abord, partant du constat que les modèles Transformeurs sont très peu exploités dans le monde non-supervisé, nous commençons par explorer leur potentiel au niveau des multiples représentations mot par mot qu'ils fournissent. Pour cela, nous établissons une méthodologie empirique et exploratoire qui nous permet de (i) mieux comprendre le fonctionnement des boites-noires Transformeurs couche par couche, ce qui est un enjeu majeur dans la recherche dans le domaine, et (ii) avoir une première idée des performances attendues dans un contexte non-supervisé.

Par la suite, après avoir étudié les similitudes et les différences entre les différentes couches d'un même modèle Transformeur, nous nous attelons à explorer la complémentarité de ces différentes couches. Pour cela, nous comparons différentes techniques dites *multiway* et qui permettent de calculer une matrice de données $\mathbf{X}$ en compressant de façon simultanée les différentes matrices $\mathbf{X}_\ell$ issues de chaque couche ℓ. Cette matrice consensus est ensuite utilisée pour réaliser du *clustering* de mots. Nous montrons que certaines de ces méthodes sont capables d'effectuer des regroupements de mots de manière efficace et interprétable. Nous évaluons leurs performances sur une grande variété de modèles Transformeur, de jeux de données, de techniques multiblocs et de méthodes de décomposition de tenseur couramment utilisées pour traiter les données qui se présentent sous forme de tables multiples. Les résultats encourageants obtenus indiquent que chaque couche apporte, en effet, quelque chose d'unique et d'utile à la tâche de clustering.

Nous nous intéressons ensuite aux représentations de phrases et de documents issues des modèles Transformeurs. Nous nous basons sur de nombreux constats posés dans de précédentes études dont la nôtre, pour émettre l'hypothèse que les modèles multicouches sont très souvent sous-exploités car seule la dernière couche est généralement utilisée. Toujours dans un contexte non-supervisé, nous proposons une nouvelle méthodologie permettant d'exploiter au mieux les modèles Transformeurs sans recourir à aucune sorte de réentraînement. Nous démontrons notamment l'intérêt de l'approche ensembliste pour (i) améliorer la qualité et la robustesse du clustering de documents tout en exploitant pleinement les modèles Transformeurs multicouches, et (ii) s'affranchir non seulement du choix de la couche à utiliser qui est très difficile dans un contexte non-supervisé, mais aussi du nombre de classes, qui est à ce jour, un problème encore ouvert.

Dans la même lignée, nous nous intéressons de manière plus approfondie aux modèles Transformeurs et à leur application au clustering et à la visualisation de données. Pour cela, nous nous focalisons sur les méthodes de transfert d'apprentissage qui consistent à réapprendre les modèles Transformeurs pré-entraînés sur une tout autre tâche afin d'améliorer

leur qualité sur des tâches futures appliquées à des données que le modèle n'aura jamais vues. Malgré la popularité de ces méthodes, elles n'ont quasiment jamais été évaluées sur une tâche de clustering alors qu'elles s'y prêtent très bien sur le plan théorique. Nous démontrons par le biais d'une étude empirique sur un large éventail de modèles que des méthodes de post-traitement basées sur la réduction de dimension sont nettement plus avantageuses que les stratégies de réapprentissage proposées en grand nombre dans la littérature. Ensuite, afin d'essayer d'expliquer les performances obtenues par les différentes méthodes de post-traitement, nous nous intéressons à la notion d'anisotropie, largement abordée dans la littérature et suspectée de réduire l'expressivité des représentations textuelles. Cette hypothèse vient du fait que les espaces de représentation fortement anisotropiques présentent des vecteurs de mots qui n'occupent qu'un cône étroit dans l'espace, ce qui implique notamment une distance cosinus très resteinte entre les paires de vecteurs de mots. Cela a conduit à plusieurs tentatives pour contrer ce phénomène à la fois sur des représentations textuelles statiques et contextuelles. Cependant, malgré cet effort, il n'y a pas de relation établie entre l'anisotropie et la performance. Afin d'y remédier, nous nous appuyons sur la tâche de classification automatique comme moyen d'évaluer la capacité des représentations textuelles à produire des classes pertinentes. Ainsi, de façon surprenante, nous montrons empiriquement un impact plus que limité de l'anisotropie sur l'expressivité des représentations textuelles à la fois en termes de directions et de similarité L_2.

Enfin, dans la dernière partie de cette thèse, nous nous intéressons à une autre problématique non-supervisée : la détection d'anomalies appliquée à la détection de contenus de formation non-éligible ou frauduleux dans le cadre du programme du compte personnel de formation (CPF). Pour cela, nous combinons les capacités qu'ont les plongements préentraînés à capturer les aspects sémantiques à partir de textes courts ainsi que les modèles de mélange, très utiles pour modéliser des groupes de données de tailles, de volumes et de formes différentes. Nous proposons un nouveau cadre de détection de contenus textuels anormaux écrits en français, adapté à deux cas que nous avons identifiés : le cas où les données concernent une thématique bien précise ainsi que le cas où les données présentent plusieurs sous-thématiques, dont le nombre n'est pas forcément connu à l'avance. Dans les deux cas, nous obtenons des résultats supérieurs à l'état de l'art, avec un temps de calcul nettement inférieur.

Cette partie répond à une problématique industrielle qui est la détection d'anomalies dans les contenus de formation de la plateforme moncompteformation (https://www.moncompteformation.gouv.fr/). En effet, au cours des dernières années, de nombreux efforts ont été consacrés à la lutte contre la fraude dans le CPF, en utilisant plusieurs types de données telles que l'évolution temporelle du chiffre d'affaires des organismes de

formation, les interactions entre les usagers et les organismes de formation, etc. Un type important d'anomalies est lorsque les organismes de formation proposent des contenus de formation factices ou non-éligibles. Nous incluons dans ce contexte les formations qui sont classées dans la mauvaise catégorie, intentionnellement ou non. Ce travail fait partie d'un projet avec des objectifs à court et à long terme. À court terme, notre objectif est d'identifier dans la base de données les contenus de formation atypiques qui peuvent être soit faux, soit inéligibles, soit mal catégorisés, afin que les formations correspondantes soient corrigées par les organismes de formation ou désactivées. À plus long terme, l'objectif est d'évoluer vers un système de détection d'anomalies en temps réel qui sera intégré dans la phase au cours de laquelle un organisme de formation renseigne les informations sur la formation proposée. Cela aidera à prévenir l'apparition d'anomalies dans la base de données, ce qui empêchera par conséquent les utilisateurs de la plateforme de tomber sur des formations qui ne devraient pas s'y trouver. C'est cet objectif à long terme qui nous motive à accorder une attention particulière au temps de calcul, car le système de détection d'anomalies doit être capable de fournir des sorties en temps réel dans un court laps de temps.

Contents

Introduction

This thesis sits at the intersection between two domains: text-mining and natural language processing. Text-mining seeks to extract meaningful information and discover structures of interest in text data, while natural language processing (NLP) is focused on enabling computers to understand and manipulate human language. The two disciplines are interconnected, each one finding application into the other. For example, NLP techniques such as word sense disambiguation and semantic parsing could see significant improvements through typical unsupervised text-mining approaches like text clustering as shown in (Kok and Domingos, 2008; Niu et al., 2004; Pantel and Lin, 2002). In the same way, existing clustering techniques can further be enhanced using common NLP methods such as part-of-speech tagging and stemming (Baghel and Dhir, 2010; Sedding and Kazakov, 2004), as part of the pre-processing phase. Both approaches of combining these two important and complementary fields are considered throughout this thesis.

Context and Motivation

The present work is made in collaboration between "Université Paris Cité - Centre Borelli UMR9010" and "Caisse des Dépôts et Consignations" (CDC) and aims to achieve both applied and academic outcomes, with a focus on addressing real-world industrial challenges and advancing the state of knowledge in the field. In particular, CDC is a general interest public institution that has a wide field of action and, like many other organizations, each of its activities generates a great quantity of text data. Examples of such corpora include (1) mails that need to be directed to the appropriate service according to their topic, (2) commercial reports that require to be organized and analyzed in order to detect various signals like complaints and suggestions, and (3) audit reports that need to be classified as critical or noncritical; those being only examples of use cases involving text data, among many other.

However, the vast majority of the available corpora are unlabeled and still need to be exploited to serve diverse purposes. A good example would be the data originating from the

public training[1] platform *MonCompteFormation*[2], where hundreds of thousands of training sessions are proposed and millions of users are registered. This leads to great volumes of data utilized in several use cases such as content-based recommendation, skills matching, course retrieval, and fraud detection.

Moreover, those corpora are of different sizes and contain sequences of different lengths, from a few words such as titles and skills to hundreds of words as in course and certification content descriptions. In the case of small corpora of short texts, it can be very challenging to represent the data samples by vector representations that accurately grasp the meaning of each text. Modern text representation models address this kind of issue by providing massively pre-trained models than can be applied to any downstream task. Those appealing methods have garnered the interest of both researchers and industrialists by their promising performance results but are still considered black boxes (Yun et al., 2021), full of mysteries. This is despite the large number of studies devoted to dissecting these models (Clark et al., 2019; Rogers et al., 2020; Tenney et al., 2019b) and that contributed to lifting the veil on how pre-trained neural language models work. However, we observed two main gaps in the overall trend of these studies which are (1) the fact that most of them focus on BERT, neglecting other similar variants, and (2) the lack of knowledge about their adequacy for unsupervised learning tasks. One of the objectives of this work is to contribute to filling these gaps.

The unsupervised learning paradigm has a long history within the field of machine learning and consists in learning useful patterns in data without explicit supervision like class labels, rating scores, pair-wise constraints, etc. This gives rise to several challenges depending on the task such as determining the optimal number of groups, choosing the most meaningful features, and setting the hyperparameters to their optimal values. Accordingly, unsupervised systems should ideally not involve any parameterization or at least keep it to a minimum and ensure that it does not have a critical impact on the outputted results. This is unfortunately not always the case, which results in techniques that are difficult or even impossible to use in real-world applications. This underscores the importance of considering both the academic and industrial perspectives, as it enables a more comprehensive appreciation of real-world challenges. In this work, we further discuss this topic and show that it is possible to implement effective unsupervised tools without requiring unrealistic parameterization.

[1] not to be confused with the machine learning meaning of "training".
[2] https://www.moncompteformation.gouv.fr/

Contributions

As mentioned above, although dense text representations are gaining in popularity, their uptake in the unsupervised learning community is very limited. This work contributes to closing this gap through different proposed methodologies that revolve around four main challenges, described as follows:

Shed light on the black-box (Chapters 2 and 3)

The first contribution is part of a continuum of research that tries to decipher popular neural-based text representations such as BERT. The particularity of our methodology is that it is exclusively based on data-mining approaches that help us deepen our understanding of Transformer-based models from a different perspective. We specifically make use of partitioning and hierarchical clustering techniques as well as multiway and tensor-based factorial analysis, basing our approach both on exploratory and performance considerations. We show through our experiments similarities, differences and more importantly, a certain complementarity between the different layers, which will be more thoroughly harnessed in the next contributions.

Efficient clustering via Transformer models (Chapter 4)

The second contribution deals with a document-level task which is document clustering. The same assessment can be reached in this regard: we do not know much about the expected performance of dense word representations in the clustering task. Moreover, in view of our previous findings, each layer of the Transformer-based language models seems to deliver something unique despite their similarity. Looking for the best layer to use is therefore pointless in addition to being impossible in the absence of labeled data. Thus, we prove that combining the layers achieves better results than conventional strategies such as using the last or the second to last layer. More importantly, we show that our proposed approach based on an *ensemble method* outperforms the use of the best layer which cannot even be accurately determined for each dataset in an unsupervised context.

Document clustering and the impact of fine-tuning language models (Chapter 5)

In order to further improve the quality of the text representations outputted by Transformer-based language models, several fine-tuning strategies have been proposed. The purpose of such approaches is to provide more transferable text representations, that can be used on various downstream tasks, especially when fine-tuning is not possible, which is typically the

case in unsupervised tasks like document clustering. However, although those approaches are designed to be tailored to all kinds of unsupervised tasks, their effectiveness has only been assessed on short text similarity. In our third contribution, the primary focus is on verifying if those popular methods are well suited for clustering long text sequences. Subsequently, we show how post-processing approaches based on dimension reduction may prove to be significantly more effective than different kinds of fine-tuning strategies.

Semantic anomaly detection in multi-topic corpora (Chapter 6)

The previous contributions were devoted to gaining insight into how to best use popular dense text representation in an unsupervised context, with an application to text clustering. This last contribution addresses a real-world need which is fraud detection in training contents with an important time efficiency constraint. To this end, we propose an anomaly detection framework in which we simulate different scenarios. In the first scenario, we assume that the data samples address one global topic, from which we try to identify deviating samples. The second scenario assumes that more than one subject is potentially present in the corpus, the aim being to identify the text samples that do not cover any of the main underlying themes. To answer both cases, we propose a fully unsupervised methodology based on mixture models and pre-trained semantic representations in which we suppose the number of groups unknown. An empirical study conclusively demonstrates that our proposal significantly outperforms state-of-the-art approaches in terms of performance and computational efficiency.

Chapter 1

Data Mining and Representation Learning: A State-of-the-Art Overview

In this chapter, we present a review of the literature to which the next chapters relate i.e. unsupervised learning applied to textual data. We first discuss text representation (or vectorization), necessary in order to handle text data. Then, we provide a short overview of existing approaches to clustering, anomaly detection and dimension reduction, with examples of real-world applications.

1.1 Text Representation

In this section, we discuss how to mathematically represent text sequences in order to be processed by any machine learning algorithm. This step is indispensable regardless of the sought task (classification, information retrieval, clustering, anomaly detection, ...) and can be carried out in several ways. The most classical way of representing a text corpus is using a document-term matrix, commonly called bag-of-words (BOW). It consists in building a matrix $\mathbf{X}$ of n rows and v columns where n is the number of documents and v the total number of unique words present in the corpus (vocabulary size). Each value x_{ij} of the BOW matrix usually represents the number of occurrences of the jth word in the ith document, which leads to a high sparsity of $\mathbf{X}$. This way of representing text is used as input to different tasks such as recommender systems (Pazzani and Billsus, 2007), question answering (Li and King, 2010), document clustering (Xu et al., 2003) and anomaly detection (Kannan et al., 2017). A common way to weight the values of the BOW matrix is TF-IDF (Term Frequency - Inverse Document Frequency) which consists in giving high importance to frequent words at the document level (TF) and low importance to frequent words at the corpus level (IDF). The advantage of the BOW representation is its simplicity since it does

not require any external information. It is also particularly useful when dealing with a corpus that is domain-specific or written in a low-resource language. The main drawback of the bag-of-word representation is that it fails at capturing underlying semantics, especially when dealing with short texts. It remains suitable as input to techniques that do not rely on semantic information and that properly handle sparse data such as co-clustering methods (Ailem et al., 2017a; Dhillon et al., 2003; Salah and Nadif, 2019) that aim at grouping rows and columns simultaneously. Otherwise, several attempts have been made to inject more semantics into the BOW like the integration of ontology representations (Hotho et al., 2001) or the use of linear and non-linear dimension reduction to embed the documents into a low-dimensional space using Singular Value Decomposition (SVD) as in (Deerwester et al., 1990) and spectral decomposition as in (He et al., 2004).

1.1.1 Word Embeddings

Transforming words into fixed-size continuous vectors that capture advanced language properties is called word embeddings. Most of the word embedding techniques are unsupervised (self-supervised), which means that they learn the syntax and semantics of a given language based on large corpora, hence providing general and task-independent representations (Radford et al., 2019). Word embeddings are often divided into two categories: prediction models and count-based models, both relying on the hypothesis that words occurring in the same context tend to have the same meaning (Harris, 1954).

Predictive models aim to learn word vectors as parameters optimizing a given objective function. This is usually achieved by extensive pre-training of language models (Bengio et al., 2000), after deriving supervision from word sequences. For example, given a sequence of s words $w_1, \ldots, w_s$, the CBOW model (Mikolov et al., 2013a) is trained to predict a word w_t given its context in a sentence as a classification task through a log-linear classifier. The hidden weights of the fully trained network are then used as word representations. In the same paper, the authors proposed skip-gram which in contrast to CBOW, predicts the surrounding context given a center word by maximizing:

$$\frac{1}{s}\sum_{t=1}^{s}\sum_{\substack{c=t-m \\ c\neq t}}^{t+m} \log p(w_c|w_t)$$

where m is the size of the context window, which means that the m words before and after w_t form its context. Skip-gram is originally stated as $m \times 2$ classification tasks, thus relying on the softmax function to determine the probability of each word within

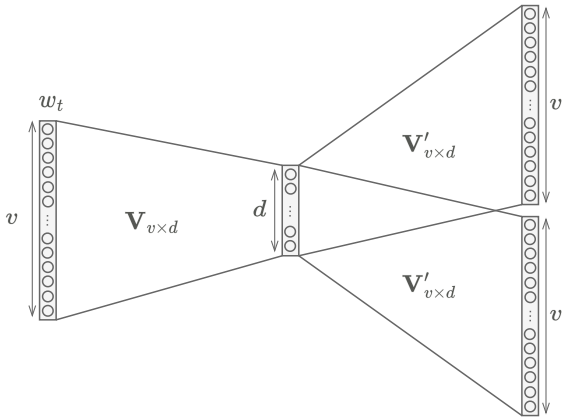

Figure 1.1: Illustration of the input and output representations in a word2vec skipgram model with window size $m = 1$, meaning that the model tries to predict one word before and one word after the center word w_t. The input is a one-hot vector of size v.

the context: $p(w_c|w_t) = e^{\delta(w_t,w_c)} / \sum_{w\in\mathcal{V}} e^{\delta(w_t,w)}$, where $\mathcal{V}$ is the vocabulary set of size v and $\delta(w_t, w_c) = \mathbf{v}_{w_t} \cdot \mathbf{v}'_{w_c}$, $\mathbf{v}_w$ and $\mathbf{v}'_w$ being respectively the "input" and "output" vector representations of the word w, part of the matrices $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_v]^\top$ and $\mathbf{V}' = [\mathbf{v}'_1, \ldots, \mathbf{v}'_v]$ represented in Figure 1.1. The problem with this formulation is that it is impractical in view of the massive number of classes that is equal to the number of words in the vocabulary. To overcome this issue, two strategies were adopted by Mikolov et al. (2013b), namely hierarchical softmax (Morin and Bengio, 2005) and negative sampling, the latter being more commonly used. Negative sampling consists in transforming the problem of v-class classification into a small number of binary logistic classifiers that predict whether a word is positive (context word) or negative (randomly sampled). Assuming m' is the number of negative samples and m the context window size, the problem becomes equivalent to taking $2m + m'$ logistic decisions, leading to the following objective:

$$\underset{\mathbf{V},\,\mathbf{V}'}{\text{minimize}} \sum_{t=1}^{s} \left[\sum_{\substack{c=t-m \\ c\neq t}}^{t+m} \log(1 + e^{-\delta(w_t,w_c)}) + \sum_{w_n\in\mathcal{N}_t} \log(1 + e^{\delta(w_t,w_n)}) \right] \tag{1.1}$$

where $\mathcal{N}_t$ is the set of negative samples. This significantly lowers the complexity of skip-gram, making "skip-gram with negative sampling" (SGNC) the most popular version of word2vec models (Mikolov et al., 2013a,b). Once the SGNC model is trained, the inner vectors $\mathbf{v}_w$ are usually used to represent each word w, although the outer vectors $\mathbf{v}'_w$ have shown interesting properties as well and have been successfully used in combination with the inner vectors (Garten et al., 2015).

SGNC has also inspired Bojanowski et al. (2017) who proposed fastText that incorporates morphological considerations into the learned representations via the n-gram sub-word information. This choice is motivated by the idea that inflected languages such as Russian and Finnish contain a large part of words with shared sub-words (roots, suffixes, etc). This is in line with the aspiration of the authors to provide ready-to-use representations for a wide range of languages (Grave et al., 2018), including low-resource languages like Kurdish and Nahuat, both highly inflected. In fastText, Equation 1.1 is optimized where the dot product is computed over all the sub-words of the center word. $\delta(w_t, w_c)$ hence becomes:

$$\delta(w_t, w_c) = \sum_{w_g \in \mathcal{G}_{w_t}} \mathbf{z}_{w_g} \cdot \mathbf{v}_{w_c}$$

where $\mathcal{G}_{w_t}$ is the set of n-gram sub-words appearing in the word w_t and $\mathbf{z}_{w_g}$ the representation of the sub-word w_g.

Count-based models in contrast to predictive models, do not rely on language modeling to produce word embeddings, but rather on count matrices that provide statistics about each word (Turney and Pantel, 2010). Various count matrices have been used in the literature including word-context and word-word co-occurrence matrices. Word-context matrices count the number of occurrences c_{ij} of a given word i in a sequence j referred to as *context*. Document-term matrices may be considered as word-context matrices (Turney and Pantel, 2010) from which it is possible to derive word representations via the LSA/LSI (Deerwester et al., 1990; Dumais et al., 1988) considering word-wise dimension reduction instead of documents as mentioned earlier. Lebret and Collobert (2014) argued that the Hellinger metric, which measures the dissimilarity between probability distributions, is better suited to deal with word-context representations than the classical Euclidean distance, inherently used in PCA. Thus, they prefer relying on Hellinger PCA to compute dense word representations achieving promising results on several NLP downstream tasks. Word-word co-occurrence matrices, on the other hand, contain pair-wise statistics between words and are commonly used to compute word embeddings via matrix factorization. In this spirit, Lund and Burgess (1996) proposed Hyperspace Analogue to Language (HAL) which uses a weighted version of word-word co-occurrence matrices where each value is weighted according to the distance between the target word and the context word. Rohde et al. (2006) analyzed how high-frequency words influence the representations provided by HAL, leading to biased similarity results. Several weighting strategies have been proposed to overcome this issue such as the use of the Pearson correlation measure (Rohde et al., 2006), frequency-based normalization (Shaoul and Westbury, 2006) and point-wise

mutual information (PMI) (Bullinaria and Levy, 2007). It is also worth mentioning that links have been established between count-based methods and predictive models by Levy and Goldberg (2014) who brought to light a connection between SGNS and the PMI word-word co-occurrence matrix factorization (shifted by a constant) and report comparable results on word similarity with a simple SVD applied to the said matrix.

Halfway between count-based and predictive models, the well-known GloVe model (Pennington et al., 2014) is based on an objective defined as a factorization of the log-count matrix. It is sometimes associated with predictive models (Levy et al., 2015) because it learns word representations by predicting the co-occurrence probabilities of words in a large training corpus. Concretely, GloVe minimizes the weighted squared difference between the dot product similarity of two word vectors and their co-occurrence log-probability. The optimized objective is hence formulated as follows:

$$\underset{\mathbf{v}_w, \mathbf{v}'_w, \forall w \in \mathcal{V}}{\text{minimize}} \sum_{t,c=1}^{|\mathcal{V}|} f(x_{ij})(\mathbf{v}_{w_t} \cdot \mathbf{v}'_{w_c} + b_{w_t} + b'_{w_c} - \log(x_{ij}))^2$$

where b_{w_t} and b'_{w_c} are the input and output bias terms. The weighting function f has been chosen to ensure that rare and frequent word pairs have a relatively lesser impact:

$$f(x) = \begin{cases} (x/x_{\text{max}})^\alpha & \text{if } x < x_{\text{max}} \\ 1 & \text{if } x \geq x_{\text{max}} \end{cases}$$

where x_{max} is set to 100 and α to 0.75 in (Pennington et al., 2014). As with skip-gram, a log-linear model is used to optimize GloVe, based on word pairs occurring within the same context window.

1.1.2 From Word to Sequence Embeddings

Representing text sequences of variable sizes by fixed-sized semantic vectors is crucial for several tasks like question answering and sentence similarity. One way of deriving text sequence embeddings is by aggregating word-level representations, usually using the arithmetic mean operation. This is possible due to an interesting property discovered about word embeddings like word2vec and GloVe which is their ability to learn complex patterns and semantics as relationships between the learned vectors such as male-female analogies. This is indicated by the fact that linear operations on the vectors often translate to some meaningful results (Mikolov et al., 2013c), e.g. *cardiologist - heart + skin = dermatologist.*

Power means, a generalization of the arithmetic mean, have shown good results on several word embeddings in (Rücklé et al., 2018) where the benefit of concatenating word vectors derived from different embedding models into one representation has also been shown. Arora et al. (2017) proposed a weighted average of word representations followed by a "common component removal" which consists in deducting the projection on the first singular vector. This improves the results of existing pre-trained representations but works only on corpora of sufficient sizes, since the weighting strategy depends on word frequencies computed over the whole corpus and given that the SVD requires a certain number of samples to be meaningful.

A criticism often made about this kind of approach is that the word order is not considered which might lead to inaccurate results especially in demanding NLP tasks. This statement needs, however, to be nuanced in light of recent findings that have shown that the word order has only limited importance in supposedly context-aware models even on complex NLP tasks like natural language inference (NLI) and paraphrase detection (Gupta et al., 2021; Pham et al., 2021; Sinha et al., 2021). It has also been demonstrated that it is quite possible to achieve satisfying results without introducing the word order in the training model (Parikh et al., 2016). Another well-known shortcoming of static word representations is that they provide one representation for each word, regardless of the surrounding context, which may be problematic, especially in word-level tasks like word sense disambiguation. To overcome these issues, word representations are often used to initialize a neural network, which in turn learns contextual connections in a task-specific fashion. Several works have adopted this strategy using convolutional (Xu et al., 2015), recurrent (Raganato et al., 2017) and self-attentive (Ruff et al., 2019) networks, with respective applications to text clustering, word sense disambiguation and anomaly detection.

1.1.3 Contextual Embedding Models

In contextual word representation, we consider that two words appearing in different contexts should not have the same representation. Thus, the objective of contextual embeddings is to provide context-aware word representations that distinguish between the different semantic variants of the same word, thus dealing with homography and polysemy. Dedicated models have been proposed in order to compute such embeddings at different levels: word, sentence and document.

Doc2vec (Le and Mikolov, 2014), sometimes called paragraph2vec, is an extension of word2vec that learns document embeddings in two ways: PVDM which is inspired by CBOW where the input word vectors are concatenated instead of summed; and DBOW, the document level equivalent of skip-gram. In both versions, a special token is introduced

among the input words. It is called *paragraph ID* and its role is not clear beside adding supervision to the modeling of sequences. Indeed, doc2vec provides a framework that allows for training sequence representations from scratch, with the possibility to inject labels via the paragraph token and perform supervised tasks such as sentiment classification. Although the purpose of doc2vec is not to provide off-the-shelf universal models, Lau and Baldwin (2016) provided a pre-trained version of doc2vec which is intended to be used on downstream tasks and applied to unseen data.

We include in the contextual embedding family models that directly provide sequence representations instead of word representations. Such is the case of skip-thought (Kiros et al., 2015), another extension of skip-gram that encodes sentences into a fixed-sized vector space. In the same manner as static word embeddings, skip-thought gets supervision from large corpora by building a training set made of a center sequence and two context sequences (one before and one after the center sequence). Predicting the context sequences is carried out through a recurrent sequence-to-sequence (seq2seq) encoder-decoder with GRU activations (Chung et al., 2014). On a similar note, Siamese CBOW (Kenter et al., 2016) is trained to distinguish between context sentences of a given center sentence and negative examples randomly sampled from the corpus. Siamese CBOW is based on a fully connected network and offers a lighter alternative to the compute-intensive RNN-based skip-thought.

Pre-training contextual sequence embeddings may also be performed with supervised objectives via labeled data as is the case for InferSent (Conneau et al., 2017), trained on language inference on the Stanford Natural Language Inference (SNLI) datasets and CoVe (McCann et al., 2017) trained on an LSTM seq2seq machine translation task. Those approaches are based on the hypothesis that models trained on certain NLP tasks produce transferable sequence representations than can be successfully used on other downstream tasks. In this same line, Subramanian et al. (2018) proposed a multi-task seq2seq model, trained on several supervised tasks like natural language inference and constituency parsing. A combination of supervised and unsupervised learning tasks has been performed by Cer et al. (2018) who observed an improvement brought about by the supervised NLI training in addition to an unsupervised objective similar to the one used for skip-thought.

Back to word-level contextual representations, ELMo (Peters et al., 2018b) is a pre-trained bidirectional contextual embedding model made of two LSTM layers. The objective of ELMo is to provide universal representations that can be used for any downstream task. A Fine-tuning step is still performed on supervised tasks but only after freezing the model's parameters and plugging a prediction layer on top of it.

Transformer-based Language Models

Since the seminal work of Vaswani et al. (2017), self-attention has revolutionized the field of language representation learning, achieving state-of-the-art results in numerous NLP tasks, featuring enhanced parallelization and improved modeling of long-range dependencies (Devlin et al., 2019; Radford et al., 2018). Following the methodology used in (Dai and Le, 2015) and Ramachandran et al. (2017), OpenAI GPT (Radford et al., 2018) is first pre-trained on an unsupervised language modeling objective and then fine-tuned on supervised objectives like question answering and text classification. The GPT architecture is based on a left-to-right self-attention decoder, which is not optimal especially for sentence-level tasks like named entity recognition for which both past and future contextual information may be useful. Devlin et al. (2019) addressed this issue and proposed a bidirectional model based on a Transformer encoder with as many inputs as outputs (one per token). The set of tokens contains pieces of words obtained using the WordPiece tokenizer which helps drastically reduce the vocabulary size and allows for representing any sequence of characters, including unseen words. BERT optimizes two objectives:

1. Masked Language Model (MLM) which consists in predicting one or more missing tokens using the context as a v-class classification task where v is the vocabulary size (number of tokens).

2. Next Sentence Prediction (NSP), a binary classification task that consists in predicting whether two input sentences follow one another or have been drawn randomly from the training corpus. The two sentences are separated by a `[SEP]` token and the classification is performed via a softmax applied on the output corresponding to the first token, generally set as the classification token or `[CLS]`.

Like GPT, BERT is intended to be fine-tuned on task-specific objectives. For example, the special token `[CLS]` can be used to predict the class under a classification purpose, and for a part-of-speech tagging task, a classifier attached to each token output can be used to predict the morphosyntactic category.

In only a few years, a plethora of Transformer-based pre-trained variants has been proposed, amending either the objective, the training data, the architecture, the hyper-parameters or the strategy used to build the vocabulary (tokenizer). RoBERTa (Liu et al., 2019b) seeks to improve BERT through a more extensive training with a larger corpus as well as a dynamic masking strategy while the NSP objective is discarded. XLNet (Yang et al., 2019b) is based on a Transformer architecture trained on an objective named permutation language modeling (PLM) (Uria et al., 2016) that has been adapted to the bidirectional setting of XLNet. The proposed objective considers the dependency between predicted targets in

addition to the dependency with the context tokens, which is more difficult to optimize, due to the factorial number of possible permutations. To address this, a hyperparameter is introduced to adjust the number of tokens used for prediction and lower the number of permutations. ELECTRA (Clark et al., 2020) uses a Transformer-based architecture trained on the Replaced Token Detection (RTD) objective, inspired by adversarial training. Two neural networks with a similar architecture are trained on different objectives: (i) first, a small generator is trained on the MLM objective and is used to substitute tokens by credible replacements, then (ii) a discriminator is trained to identify whether the tokens are original or replaced, using the frozen parameters of the generator. Once trained, the discriminator is used as an encoder, either for generating embeddings or for fine-tuning it on supervised tasks. The Transformer architecture have also been used in encoder-decoder models that are particularly useful for seq2seq tasks like text generation and machine translation. MASS (Song et al., 2019) has been the first to propose an encoder-decoder Transformer-based model, followed by several other variants like T5 (Raffel et al., 2020) and BART (Lewis et al., 2020), both using different objectives and training corpora.

Most of the above-mentioned works provide models pre-trained on English corpora. However, similar architectures have also been trained in various languages like French (Le et al., 2020; Martin et al., 2020) and Arabic (Antoun et al., 2020). Several multilingual versions of existing models have also been proposed like XLM-R (Conneau et al., 2020) and mT5 (Xue et al., 2021), respectively trained on corpora of 100 and 101 languages. Besides, within the same language, word distributions may greatly differ from one domain to another. Considering this, existing pre-trained models have been re-trained on domain-specific corpora in order to capture more precise concepts with respect to a given field like biomedical sciences (Lee et al., 2019) and mathematics (Peng et al., 2021). However, one drawback of using general-purpose models as a starting point lies in the fact that the tokenizer used is not built from the domain-specific vocabulary in which word frequencies may be highly different. This may be overcome by training the model from scratch, which requires, however, a large amount of data and a more extensive training. Two popular examples of such approaches are FinBERT (Yang et al., 2020) dedicated to the finance domain and SciBERT (Beltagy et al., 2019) trained on a large corpus of scientific articles.

Given the large number of parameters learned by the multilayered Transformer models, several attempts have been made to reduce the computational demand of Transformer-based language models. For example, Sanh et al. (2019) proposed a compact model that is trained based on the teacher-student paradigm which consists in training a student model (DistilBERT) to mimic the behavior of a teacher model (BERT), thus retaining most of the performance of original-sized Transformer-based models. A similar approach called

XtremeDistil (Mukherjee and Hassan Awadallah, 2020) has been applied to the multilingual version of BERT, where an SVD-reduced version of the mBERT static input embeddings are used as initial embeddings. A very different strategy has been adopted by Lan et al. (2020) who proposed ALBERT, made of 18x fewer parameters than BERT while having the same number of layers and hidden dimensions. This is made possible particularly by parameter sharing, which consists in duplicating the learned parameters across layers. This certainly reduces the number of trained parameters and hence the storage memory but does not affect the inference time (Lu and MacNamee, 2020), even though a lower training time has been reported in (Lan et al., 2020). As for the training objective of ALBERT, the authors argued that the NSP objective is not challenging enough for it is partly comparable to a topic prediction task. It is hence replaced by the sentence order prediction (SOP) objective that predicts whether two consecutive sentences appear in the right order, thereby focusing on the more difficult coherence prediction task.

Pre-trained Transformer models can be used either in a feature-based way or via fine-tuning. When dealing with supervised tasks like sentiment analysis and named entity recognition, fine-tuning is recommended in order to enrich the model with task-specific insights. Otherwise, in an unsupervised setting or a situation where re-training is unfeasible, the feature-based approach remains a good alternative to produce dense representations that can be fed into various machine learning algorithms. In order to further improve the feature-based approach, fine-tuning strategies (Carlsson et al., 2021; Cheng, 2021; Gao et al., 2021; Liu et al., 2021; Reimers and Gurevych, 2019) have been proposed specifically to make better quality and more transferable sequence representations.

These are only some examples of the numerous existing Transformer-based contextual approaches, many others can be found in (Kalyan et al., 2021).

1.2 Clustering

Clustering is a major branch of the unsupervised learning domain that aims at organizing a set of objects into homogeneous groups (or clusters) so that similar objects are grouped together. The clustering task is key when dealing with unlabeled data and is utilized in a wide range of domains such as pattern recognition, recommender systems and bioinformatics. Given a set of n objects (or samples, individuals, instances), each object is commonly described by d features (or variables) and represented by a vector $\mathbf{x}_i = (x_{i1}, \dots x_{id}), i = 1, \dots, n$. The whole set of data is hence represented by a matrix $\mathbf{X}$ of size $n \times d$ and is intended to be split into g homogeneous parts.

Depending inter alia on the data used as input, many options are available regarding the clustering approach to use. In this section, we describe briefly the main families of clustering techniques as well as their specificities. For a comprehensive review of the existing clustering techniques and their applications, the reader is referred to Saxena et al. (2017) and Xu and Wunsch (2005).

1.2.1 Hierarchical Clustering

Hierarchical clustering methods structure the data samples in the form of a hierarchy, represented by a binary tree called *dendrogram*. The hierarchy structure goes from singletons to the whole set of samples (bottom-up strategy) or inversely (top-down strategy). These two strategies lead to *Agglomerative* and *Divisive* approaches respectively.

Agglomerative clustering approaches first affect each data sample into a singleton cluster then progressively merge pairs of clusters until obtaining a single group that gathers the whole set of samples. This process is carried out greedily by merging the clusters that are the closest according to a predetermined criterion that derives the dissimilarity between two clusters from the pairwise distances between their members. Any distance/similarity measure can be used such as the classical Euclidean distance and the Pearson Correlation score, usually used on genomic data (Bohlin et al., 2009). Once the pairwise distance is measured between cluster members, several criteria or linkage functions can be used in order to assess the distance between clusters. The most popular linkage functions are: single-linkage (Sibson, 1973) which is the distance between the two closest samples in the two clusters, complete-linkage (Sorensen, 1948) which considers the distance between the two most distant samples and average-linkage (Sokal, 1958) that uses the mean of the distances between the two clusters. When dealing with continuous data, another popular linkage function is Ward's criterion which minimizes the variance within the clusters to be merged (Ward Jr, 1963). Many other agglomerative approaches and variants exist in the literature like BIRCH (Zhang et al., 1996) and CURE (Guha et al., 1998). BIRCH is known for its scalability and requires only one scan of the whole data by the means of a Clustering Feature Tree (Zhang et al., 1996), which makes it suitable for large numerical datasets. CURE is an agglomerative hierarchical approach that is well suited for the clustering of noisy data. It uses only a handful of data samples (called representatives) in each cluster to determine the proximity of two clusters.

Divisive clustering approaches, as opposed to agglomerative techniques, first consider a single cluster containing all of the data samples, then proceeds to successive subdivisions until either a termination criterion is met, or each cluster is assigned to its own singleton cluster. Several strategies can be adopted for the splitting process including

monothetic approaches based on a single variable like MONA (Kaufman and Rousseeuw, 2009) and DIVCLUS-T (Chavent et al., 2007), and polythetic approaches which use all of the variables such as DIANA (Kaufman and Rousseeuw, 2009) and the Edwards and Cavalli-Sforza's method (1965). Divisive approaches are known for their significant computational complexity (Milligan and Cooper, 1987), for they have to examine an exponential number of subgroups to determine the best subdivision (Edwards and Cavalli-Sforza, 1965). This makes the divisive family of hierarchical approaches less popular compared to agglomerative hierarchical clustering (AHC).

The main advantage of hierarchical clustering approaches is that they do not require the number of clusters in advance and can provide different clustering partitions according to the level at which the dendrogram is cut. Another advantage of this family of methods is their interpretation capabilities thanks to the visualization possibilities offered by dendrograms. However, agglomerative approaches also tend to have high complexity which makes them unsuitable for large data analysis. Complete-linkage and single-linkage are though less slow-running, with a quadratic complexity, against $O(n^3)$ for other agglomerative strategies and $O(2^n)$ for divisive approaches.

1.2.2 Partitioning Approaches

This kind of method is usually opposed to hierarchical techniques and aims to structure the data into distinct clusters in a non-hierarchical way. k-means is undeniably the most popular partitioning algorithm. It is highly appreciated for its efficiency and low complexity. k-means (MacQueen et al., 1967) is an iterative algorithm that aims to find g representatives (centroids or means) of the data, one for each of the g clusters, so that the samples of the same cluster are as close as possible. This constitutes the criterion optimized (minimized) by k-means and is referred to as the Inertia or the Sum of Squared Error (SSE). It is defined as:

$$SSE = \sum_{k=1}^{g} \sum_{\mathbf{x}_i \in c_k} d(\mathbf{x}_i, \boldsymbol{\mu}_k)$$

where $\boldsymbol{\mu}_k$ is the centroid of the cluster k and c_k the set of samples belonging to the cluster k. d is a distance function that quantifies how far the sample $\mathbf{x}_i$ is from the centroid $\boldsymbol{\mu}_k$. The inertia is optimized through an iterative procedure with the following steps:

1. Initialize randomly the g centroids $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_g$.

2. Assign each sample i to the closest centroid $\boldsymbol{\mu}_h$ such that $h = \text{argmin}_k \; d(\mathbf{x}_i, \boldsymbol{\mu}_k)$.

3. Update each centroid $\boldsymbol{\mu}_k,\ k = 1, \ldots, g$ with the mean of its assigned members.

4. Repeat until the centroids do not change.

In the conventional version of k-means (MacQueen et al., 1967), the squared Euclidean distance is used but alternative distances can be used such as the cosine distance computed as:

$$d_{cos}(\mathbf{x}_i, \mathbf{x}_j) = 1 - \frac{\langle \mathbf{x}_i, \mathbf{x}_j \rangle}{\|\mathbf{x}_u\| \, \|\mathbf{x}_v\|} \tag{1.2}$$

In this version, the mean representatives are normalized to unit norm. This approach is called Spherical k-means and is particularly well suited for text clustering (Dhillon and Modha, 2001).

Other variants of k-means have been proposed such as fuzzy k-means where each data point can belong to more than one cluster, k-medoid (Rdusseeun and Kaufman, 1987) in which the centroids are updated as an actual member of the set of samples, and k-medians (Bradley et al., 1996) that considers the representative of each cluster as the median rather than the mean of its members and hence assesses the closeness of a sample to a cluster via the Manhattan (or 1-norm) distance.

1.2.3 Density-based Clustering

Density-based approaches consider clusters are high-density regions of arbitrary shapes, separated by low-density regions made of noise. The advantage of such methods is their ability to discover clusters of all shapes and sizes. The most famous density-based algorithm is DBSCAN (Ester et al., 1996) which relies on two main parameters: the radius ε and the density threshold *minPts*. A "core point" is a data point that has at least *minPts* samples around its radius, otherwise it is called a "border point". Each cluster starts with a random core point and progressively grows by adding density-reachable points and discovering new core points within the ε-neighborhood, until no progression can be made. At this point, another cluster is started with another core point, until all the data samples have been processed. The data points that have not been reached by the gathering process are considered outliers for they belong to low-density regions. The main drawback of DBSCAN is its sensitivity to the two parameters ε and *minPts*, which is problematic in the unsupervised setting of the clustering task. Another weakness of DBSCAN is its incapacity to discover clusters of different densities. OPTICS (Ankerst et al., 1999), proposed by the same authors, addresses both issues by discovering nested clusters with different degrees of density represented by a hierarchical structure called reachability-plot that corresponds to a wide range of parameter values. This makes the clustering more interactive and visual.

DBSCAN and its variants are known for their limitations regarding high dimensional data, which has led to several approaches to overcome this issue like DENCLUE (Hinneburg and Keim, 1998) that is designed for handling noisy and high dimensional data.

1.2.4 Graph-based Clustering

This family of methods includes clustering techniques that consider the input data as a graph $G = (V, E)$ of $n = |V|$ nodes and $e = |E|$ edges connecting pairs of nodes, usually represented by a squared and symmetric adjacency matrix of size $n \times n$. Graph-based clustering approaches generally aim to partition the set of nodes of an undirected graph into clusters (or communities) in such a way that the nodes within a cluster are as connected as possible whereas fewer edges connect the nodes of different clusters. A simple and popular formulation of the problem is the *minimum cut* (or *mincut*) problem which consists in finding the partition that minimizes either the number of edges or the sum of edge weights between clusters. Improved versions such as *minimum ratio cut* (Leighton and Rao, 1989) and *normalized ratio cut* (Shi and Malik, 2000) have been proposed in order to avoid one-node clusters, usually present in the *mincut* solution (Wu and Leahy, 1993).

Another well-known measure to assess the quality of graph clustering is the modularity function (Newman and Girvan, 2004) which evaluates to what extent the nodes inside each cluster are connected in comparison to an expected (or random) edge-density. Modularity measure lies between -1 and 1 for unweighted (binary) graphs and has been adapted to weighted graphs in (Newman, 2004a).

Given (i) a weighted graph G of weighted degree w and represented by an adjacency matrix $\mathbf{A}$, and (ii) a clustering partition represented by a binary membership matrix $\mathbf{Z} = (z_{ik})$ of size $n \times g$, the modularity is expressed as:

$$Q = \frac{1}{2w} \sum_{i,j=1}^{n} (a_{ij} - m_{ij})\, \sigma_{ij}$$

where $m_{ij} = \dfrac{w_i w_j}{2w}$ is the expected density between the nodes i and j of weighted degrees $w_i = a_{i.}$ and $w_j = a_{j.}$ respectively. $\sigma_{ij} = \sum_{k=1}^{g} z_{ik} z_{jk}$ equals one when the nodes i and j are assigned to the same clusters and zero otherwise. The modularity criterion can be optimized w.r.t. $\mathbf{Z}$ in different ways like spectral decomposition (Newman, 2006; White and Smyth), mathematical programming (Agarwal and Kempe, 2008), and meta-heuristics (Duch and Arenas, 2005; Nicosia et al., 2009; Reichardt and Bornholdt, 2006). Fast greedy approaches have also shown their effectiveness on large graphs (Clauset et al., 2004; Newman, 2004b), one of the best-known being the Louvain algorithm (Blondel et al.,

2008) based on a relatively simple heuristic that finds a hierarchical community structure. As an agglomerative approach, the Louvain algorithm first assigns each node to its own community. Then the two following phases are alternated until a local optimum of the modularity is reached:

- **Phase 1:** place each node i in the neighboring community C that maximizes the gain of modularity:

$$\Delta Q = \left[\frac{\sum_{in}+w_{i,in}}{2w} - \left(\frac{\sum_{tot}+w_i}{2w}\right)^2\right] - \left[\frac{\sum_{in}}{2w} - \left(\frac{\sum_{tot}}{2w}\right)^2 - \left(\frac{w_i}{2w}\right)^2\right]$$

  where $\sum_{tot}$ is the sum of all the edge weights of the nodes in C, $\sum_{in}$ is the sum of the edge weights that connect the nodes inside the community C, and $w_{i,in}$ is the sum of the weights of the edges that connect the node i with the nodes that belong to C. This step is repeated until no further improvement can be made.

- **Phase 2:** build a smaller graph where each community obtained in Phase 1 becomes a node and the edge weight between two nodes is computed as the sum of the edge weights connecting the corresponding two communities.

The advantage of Louvain is its low complexity which is mainly due to the simpleness of the expression of ΔQ. Another advantage of Louvain is its hierarchical scheme that may be helpful when the number of communities is unknown.

Graph clustering may be applied to any type of data, provided that an adjacency matrix can be derived from the original data representation. In this view, spectral clustering (Von Luxburg, 2007) may be included in the graph-based clustering family of approaches since it works on graph representations of the data, like the graph Laplacian matrix, usually compressed via spectral decomposition then fed into a clustering algorithm like k-means (Ng et al., 2001; Shi and Malik, 2000). These kinds of methods are well known for their ability to discover groups of arbitrary shapes since they make few assumptions on cluster shapes, which constitutes a considerable advantage when dealing with real-world datasets.

Graph-based algorithms may also be used on bipartite graphs represented by a matrix of size $n \times n'$ with the purpose of finding bi-clusters or co-clusters, i.e. row and column partitions. This kind of approach is usually proposed for document clustering on sparse text representations, considering the documents and the words as two distinct node sets of the same bipartite graph (Ailem et al., 2015; Dhillon, 2001; Labiod and Nadif, 2011).

1.2.5 Model-based Clustering

Finite mixture models are probabilistic approaches for data clustering and assume that each group k is generated according to a probability distribution of density function φ with different parameters θ_k. Given a data matrix $\mathbf{X}$ of size $n \times d$, the probability density function of the mixture of g models is expressed as follows:

$$f(\mathbf{X}; \Theta) = \prod_{i=1}^{n} \sum_{k=1}^{g} \pi_k \varphi(\mathbf{x}_i | \theta_k)$$

where $\Theta = (\pi_1, \ldots, \pi_g, \theta_1, \ldots, \theta_g)$ contains the parameters of the mixture, $\varphi(\mathbf{x}_i|\theta_k)$ is the density function for observation $\mathbf{x}_i$ with parameters θ_k, π_k is the probability or proportion of the kth component (or distribution) such that $\pi_k > 0,\ \forall k = 1, \ldots, g$ and $\sum_{k=1}^{g} \pi_k = 1$. A mixture of g models assumes that the data samples $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are randomly drawn according to the following generative process:

- Choose a component $k \sim \text{Multinomial}(\pi_1, \ldots, \pi_g)$.
- Choose a data sample $\mathbf{x}_i \sim \varphi(\mathbf{x}_i|\theta_k)$.

In order to estimate the parameter of the generative model, the Maximum Likelihood approach is commonly used. It consists is finding the parameters Θ that maximizes the likelihood of the observed data. In other words, the goal is to find the model in which the observed data is most probable. Given the difficulty of directly optimizing the likelihood function, the log-likelihood is maximized instead and is written as:

$$L(\mathbf{X}; \Theta) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{g} \pi_k \varphi(\mathbf{x}_i | \theta_k) \right)$$

This task is commonly performed via the iterative Expectation-Maximization (EM) algorithm (Dempster et al., 1977) and can be applied to different families of distributions. One of the most popular model-based approaches for continuous data is the Gaussian Mixture Model (Banfield and Raftery, 1993; Celeux and Govaert, 1995), which assumes the data samples to be drawn according to g Gaussian distributions (cf. Section 6.3 for more details). According to the data type and the hypotheses that can be made, different distributions can be used such as Bernoulli for binary data (McCallum et al., 1998; Nadif and Govaert, 1997), Poisson for count data and contingency tables (Boutalbi et al., 2021; Brijs et al., 2004; Karlis and Meligkotsidou, 2007; Riverain et al., 2022a,b), Multinomial for categorical data (Jollois and Nadif, 2002), and von Mises-Fisher for directional data (Affeldt et al., 2021; Banerjee et al., 2005; Gopal and Yang, 2014; Salah and Nadif, 2017b).

The power of mixture models lies in their flexibility as they are able to deal with specific situations like noisy, heterogeneous, unbalanced or differently shaped data. Under certain assumptions like the shape and size of the clusters, some well-known algorithms can be derived from mixture models. For instance, k-means corresponds to a restrictive mixture of Gaussian distributions, where the clusters are constrained to be spherical, equally sized and of equal variance, which translates mathematically into equal spherical covariance matrices Σ_k (cf. Section 6.3) as well as equal proportions $\pi_k,\ k = 1, \ldots, g$. For further detail about model-based clustering approaches, the reader is referred to McLachlan and Peel (2004) and Govaert (2009).

1.2.6 Applications

Cluster analysis is a powerful toolbox made available to practitioners in order to synthesize and create value from huge amounts of data. Clustering techniques can also be used to improve other machine learning tasks such as text classification (Baker and McCallum, 1998), information retrieval (Anick and Vaithyanathan, 1997), image segmentation (Pappas and Jayant, 1989) and outlier detection (Jiang et al., 2001).

Many industrial and academic fields also make use of cluster analysis, for different purposes. For instance, clustering is commonly used in the scientific community like biomedicine (Yoo et al., 2012) and molecular biology (Gao et al., 2006). In particular, gene expression microarray data are generally represented by high dimensional numeric matrices in which each entry represents the expression level of a gene (row) under a given condition (column). Cluster analysis is commonly used on this kind of data in order to discover underlying patterns and coexpression of genes and help identify co-regulated or functionally related genes (Ben-Dor et al., 2002). Clustering approaches are also often used to organize articles in the academic literature (Aljaber et al., 2010) such as biomedical articles (Yoo and Hu, 2006). On the same note, text data clustering has benefited several other domains like forensics (Decherchi et al., 2009) and psychiatry (Abbe et al., 2016). More specifically, topic modeling approaches like LDA and NMF have been widely used to cluster words, sentences and documents with applications like web semantic tagging (Allahyari and Kochut, 2016), web services analysis (Aznag et al., 2013), email surveillance (Berry and Browne, 2005), biomedical information retrieval (Chen et al., 2012).

Deep clustering has also been exploited to analyze various sources of real-world data. For instance, deep architectures have often been leveraged to perform clustering on medical imaging (Kart et al., 2021; Mittal et al., 2021; Moriya et al., 2018), single-cell RNA-seq data (Brbić et al., 2020; Tian et al., 2019) and textual data (Hadifar et al., 2019; Nadif and Role, 2021; Xu et al., 2017).

1.2.7 Clustering Validation

Several strategies exist in order to assess the quality of a clustering partition. Some of them do not require any external information about the group structure and are called *internal measures*. They quantify the cohesion between the samples of the same group (compactness) and the distance between groups (separation). Such is the case, for instance, of the silhouette index (Rousseeuw, 1987), Dunn's index (Dunn, 1974) and Davies-Bouldin index (Davies and Bouldin, 1979). For further details about internal indices, please refer to Charrad et al. (2014); Lamirel et al. (2016).

In a purpose of performance evaluation and comparison, it is common to rely on *external measures* that require to know the ground-truth labels. These measures are analogous to those used in supervised classification and consist in assessing to what extent the partition provided by a clustering algorithm matches the given labels. The best-known external measures include the Normalized Mutual Information (NMI) (Strehl and Ghosh, 2002), the Adjusted Rand Index (ARI) (Hubert and Arabie, 1985; Steinley, 2004) and clustering accuracy. The NMI score represents the mutual information between two partitions A and B, normalized by their entropies. It is computed as:

$$\text{NMI}(A, B) = \frac{\text{MI}(A, B)}{\sqrt{\mathcal{H}(A)\,\mathcal{H}(B)}}$$

where $\mathcal{H}(Y)$ is the entropy defined by $\sum_{i=1}^{|Y|} p(y_i) \log p(y_i)$ where $|.|$ denotes the cardinality, and MI is the mutual information, defined as:

$$\text{MI}(A, B) = \sum_{i=1}^{|A|} \sum_{j=1}^{|B|} p(i, j) \log \frac{p(i, j)}{p(i)\,p(j)}$$

The Rand Index (RI) is another external measure based on the agreements and disagreements between partitions. Given two partitions of size c, RI is computed as $(c_{00} + c_{11})/\binom{c}{2}$ and its adjusted version (ARI) as:

$$\text{ARI}(A, B) = \frac{2(c_{00}c_{11} - c_{01}c_{10})}{(c_{00} + c_{01})(c_{01} + c_{11}) + (c_{00} + c_{10})(c_{10} + c_{11})}$$

where $c_{00} + c_{11}$ is the number of cases where the partitions agree and $c_{01} + c_{10}$ the number of cases where they disagree. The binomial coefficient $\binom{u}{v}$ can be interpreted as the number of ways to choose u elements from a v-elements set.

Another popular external measure is the clustering accuracy, similar to the one used in supervised classification, adapted to the unsupervised setting of clustering. It consists in

computing the accuracy after relabelling the evaluated partition which consists in finding the best match between the provided true classes and the evaluated clusters. It is computed as follows:

$$\text{ACC} = \frac{1}{n} \max \left[\sum_{\mathcal{G}_k, \mathcal{L}_m} T(\mathcal{G}_k, \mathcal{L}_m) \right]$$

where $\mathcal{G}_k$ is the kth cluster in the evaluated partition and $\mathcal{L}_m$ is the true mth class. $T(\mathcal{G}_k, \mathcal{L}_m)$ is the number of entities that are assigned to cluster k while belonging to class m. Caution must be taken when using this measure since it presents the same drawback as the classification accuracy, which is that it is sensitive to data imbalance. It is hence recommended to use it in conjunction with other measures or to accompany the accuracy score by a confusion matrix.

1.3 Dimension Reduction

Dealing with high-dimensional data has become commonplace in several domains like molecular biology (Stears et al., 2003), astrophysics (Richards et al., 2009) and heathcare (Zhu et al., 2016). In some cases, the number of features attains and even exceeds the number of observations, which leads to the well-known *curse of dimensionality* problem, first stated by Bellman (1961) as the fact that the need for learning samples grows exponentially with the dimensionality. In such cases, a significant part of the features is usually either redundant (presence of correlation with other features), or noisy (variables that do not bring any useful information), or poorly discriminatory (variables that do not help the modeling of data); as a consequence, the meaningful features are buried under the overwhelming presence of irrelevant information. Given a dataset $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ of n samples $\mathbf{x}_i \in \mathbb{R}^d$, d being the number features, the objective of dimension reduction is to find a lower-dimensional representation $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ where $\forall i = 1, \ldots, n,\ \mathbf{y}_i \in \mathbb{R}^{d'}$ with $d' < d$, while preserving the maximum of the information present in the original data (Fodor, 2002). To address this problem, it is possible either to filter the features in order to keep only the most relevant ones, or represent the observations in a whole new reduced space in which each data point is described by a small number of dimensions.

1.3.1 Feature Selection

Feature or variable selection consists in eliminating the irrelevant features according to a given criterion (Koller and Sahami, 1996; Quafafou and Boussouf, 2000). The most simple approach is to set a filtering condition that depends on what is considered irrelevant.

For example, for tackling redundancy in the features, statistical tests such as the χ^2 test, the Pearson correlation test and the Spearman correlation test can be used to discard dependant features. The family of methods to which these approaches belong is called "filter approaches" (Almuallim and Dietterich, 1994; Liu and Setiono, 1996) which includes techniques that are based only the input data to perform feature selection. These approaches are naturally well suited in the absence of labels, when the performance cannot be assessed. In supervised learning, the interdependence can be assessed between the explanatory features and the target variable in order to keep the most discriminatory features (Kwak and Choi, 2002; Torkkola, 2003). The feature selection problem can also be formulated as finding the best minimal subset of variables that satisfies a chosen criterion. This constitutes an NP-hard problem (Chen et al., 1997) that is usually tackled using "wrapped feature selection" approaches. These techniques consist in evaluating combinations of features using a black-box learning algorithm to find a near-optimal solution for several tasks such as pattern classification (Siedlecki and Sklansky, 1989) and clustering (Dy and Brodley, 2004).

1.3.2 Linear Dimension Reduction

The most popular linear dimension reduction method is Principal Component Analysis (PCA). It projects the data into a low-dimensional space with maximum variance and produces a linear mapping between the original variables into the reduced space. In other words, PCA finds new features that are linear combinations of the original features while capturing the maximum amount of variation within the dataset. The objective function of PCA can be written as:

$$\underset{\mathbf{W},\mathbf{W}^\top\mathbf{W}=I_{d'}}{\text{minimize}} \|\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{W}^\top\|_F^2 \tag{1.3}$$

where $\mathbf{X}$ is a centered matrix of size $(n \times d)$ and $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_d]$ can be seen as the orthogonal mapping matrix of size $(d \times d)$. It is given by the eigenvectors of the covariance matrix $\mathbf{S} = \frac{1}{n}\mathbf{X}^\top\mathbf{X}$ obtained by the eigen-decomposition of $\mathbf{S} = \mathbf{W}\mathbf{\Lambda}\mathbf{W}^\top$ where $\mathbf{\Lambda}$ is a diagonal matrix containing the eigenvalues $\lambda_1, \ldots, \lambda_d$ of $\mathbf{S}$. Note that $\mathbf{X}^\top\mathbf{X}$ becomes prohibitively large with increasing d which makes this solution unpractical. In such case, it is possible to rely on Singular Value Decomposition (SVD) that closely relates to PCA. It consists in directly decomposing $\mathbf{X}$ into $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where $\mathbf{U}$ is an orthogonal matrix of size $n \times n$ containing the left singular values of $\mathbf{X}$, $\mathbf{\Sigma}$ is a diagonal matrix of size $n \times d$ that contains the singular values of $\mathbf{X}$, $\mathbf{V}$ is of size $d \times d$ and is made of the right singular values of $\mathbf{X}$. The eigen-decomposition of $\mathbf{X}^\top\mathbf{X}$ can be deduced from the SVD as $\mathbf{W}$ is equal to $\mathbf{V}$ and each jth eigenvalue λ_j is equal to the square of the jth singular value σ_j ($\sigma_j = \sqrt{\lambda_j},\ \forall j = 1, \ldots, d$). The reduced version of $\mathbf{X}$ is hence expressed by

$\mathbf{Y} = \mathbf{X}\mathbf{W}_{d'} = \mathbf{X}\mathbf{V}_{d'} = \mathbf{U}_{d'}\mathbf{\Sigma}_{d'}$, where the index $_{d'}$ means that we take only the d' first columns.

Several variants of PCA and SVD have been proposed in order to address specific situations like the presence of outliers (Xu et al., 2010), missing values (Ilin and Raiko, 2010) or high dimensionality (Zou and Xue, 2018). Besides, other linear techniques exist such as Locality Preserving Projection (LPP) (He and Niyogi, 2003) that finds linear mapping the best preserve the neighborhood structure of the dataset, and Nonnegative Matrix Factorization (NMF) (Lee and Seung, 1999) that incorporates an additional constrain which is the nonnegativity of both the original and reduced space. NMF is also considered as a clustering technique, very popular in the text-mining community (Febrissy and Nadif, 2020; Xu et al., 2003). For more details about these and other linear methods, the reader is referred to Cunningham and Ghahramani (2015).

1.3.3 Manifold Learning

This family of methods is also referred to as nonlinear dimension reduction and is often considered as a generalization of linear techniques that is able to discover non-linear structures in high-dimensional data. For instance, Kernel-PCA (Schölkopf et al., 1998b) is an extension of the classical PCA that first maps the data into a high-dimensional space in which the data become linearly separable. The mapping is performed using a nonlinear kernel that preserves the nonlinear structure of the data. During this step, $\mathbf{X}$ is transformed into a positive semi-definite dot-product similarity matrix $\mathbf{A}$ of size $n \times n$, called the kernel matrix. Then, the projection onto a low-dimensional space is given by the eigendecomposition of $\mathbf{A}$. This makes Kernel PCA well suited to high-dimensional data since its complexity does not grow with the number of features d. The most typical kernels used include the Gaussian kernel (also called RBF for radial basis function) and the Polynomial kernel (Schölkopf et al., 1998a). There are many other popular manifold techniques such as Multidimensional Scaling (Kruskal and Wish, 1978), Laplacian Eigenmap (Belkin and Niyogi, 2001), isometric feature mapping (Isomap) (Tenenbaum et al., 2000), Locally Linear Embedding (Roweis and Saul, 2000), all of which can be shown to be special cases of Kernel PCA under specific conditions (Bengio et al., 2004; Williams, 2000).

t-Stochastic Neighbor Embedding (t-SNE) (van der Maaten and Hinton, 2008) is a manifold technique, mainly used for the 2D visualization of high-dimensional data. It is a variant of the Stochastic Neighbor Embedding (SNE) (Hinton and Roweis, 2002) and aims at minimizing the Kullback–Leibler divergence between two distributions P and Q. P represents the proximity or neighborhood of the objects in the high-dimensional space which means that the more two data points are close, the more they are assigned

a high probability. Q follows the same scheme, this time in the low-dimensional space. t-SNE is very popular for its visualization capabilities. However, due to its algorithmic limitations, it does not scale well on large data and it is not possible to go further than 3 dimensions. Another disadvantage of t-SNE is its sensitivity to hyperparameter tuning, especially perplexity (connected to the number of data points that can be considered as neighbors).

More recently, McInnes et al. (2018) have proposed Uniform Manifold Approximation and Projection (UMAP) which is, like t-SNE, based on KNN-graphs. UMAP creates a uniform embedding space which means that the radius that determines if two points are close to each other depends on the density of the region to which the points belong. To this end, the function used to compute the radius depends on how far the kth nearest neighbor lies; k being one of the most important hyperparameters of UMAP. Note that UMAP also relies on simplicial sets that can be viewed as a higher-dimensional generalization of graphs, which helps drastically lowering the computational complexity. These choices allow for a better integration of the global structure of the data into the construction of the embedding space, and are supported by solid mathematical foundations based on the Riemannian metric and topological spaces. UMAP has a few similarities with t-SNE but brings several improvements, especially via the graph construction and the loss function. In addition, UMAP may be used as a pre-processing tool since it has no restrictions regarding the number of dimensions. However, when the pre-processing precedes an unsupervised task, the problem of hyperparameter tuning arises yet again, even though UMAP is much less sensitive to the choice of the number of neighbors k and has fewer parameters to fix. Another advantage of UMAP is its linear complexity against $O(n^2)$ for t-SNE in its exact version and $O(n \log n)$ when the Barnes Hut approximation (van der Maaten, 2013) is used.

Autoencoders

Autoencoders (Hinton and Salakhutdinov, 2006; Rumelhart et al., 1985) are well-known neural techniques used to perform several tasks such as data denoising and anomaly detection. In particular, undercomplete (or bottleneck) autoencoders are commonly used for dimension reduction (Figure 1.2), providing good-quality latent representations on several types of data. We can define an autoencoder as a neural network trained to compress the input data samples into a reduced space and decompress them without losing too much information. In other words, the objective of the autoencoder is to learn a latent representation of the data from which it is possible to reconstruct a copy of the input data that is as close as possible to the original. If we use the Euclidean loss to measure the reconstruction error, we can formulate the objective as finding the functions f and g that

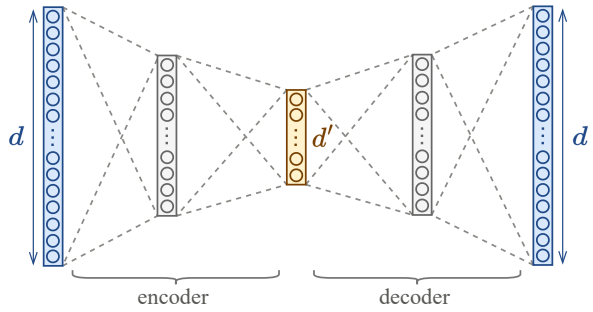

Figure 1.2: Example of a bottleneck autoencoder architecture.

minimize:

$$\frac{1}{n}\sum_{i=1}^{n} \|\mathbf{x}_i - g(f(\mathbf{x}_i))\|^2 \tag{1.4}$$

where f is the encoding function (encoder) that projects the input data into the reduced space and g is the decoding function (decoder) responsible for reconstructing the embedded data into the original dimension. Under this basic formulation (Goodfellow et al., 2016), if f and g induce linear transformations, then the optimized criterion becomes closely related to the one optimized by PCA and given by Equation 1.3 (Baldi and Hornik, 1989; Bourlard and Kamp, 1988). This is the case when both the encoder and the decoder have one single layer with linear activations. This configuration is rarely used since the power of autoencoders lies in their non-linearity. Furthermore, regularized versions of autoencoders are more usually used such as sparse autoencoders in which a regularization term is added to the objective function that encourages fewer nodes to activate for each data point. The aim of sparse autoencoders is to reduce the capacity (or freedom) of the model so that the neurons are activated only when it is most meaningful to do so. This is achieved through an L_1 penalty applied to the output of the latent hidden layer. Denoising autoencoders are another well-known variant that seeks to minimize the difference between $\mathbf{x}_i$ and $g(f(\tilde{\mathbf{x}}_i))$ where $\tilde{\mathbf{x}}_i$ is a copy of the ith data sample $\mathbf{x}_i$ contaminated with some form of noise. By doing so, denoising autoencoders learn to compress data into a reduced space while dealing with noise, which might increase the robustness of the trained model. Variational autoencoders (VAE) (Kingma and Welling, 2014) are also based on the encoding decoding scheme but have a different mathematical formulation from traditional autoencoders. Variational autoencoders belong to the family of variational Bayesian methods and, as such, learn latent representations made of a mixture of distributions. An analogy can hence be made between VAE and EM, whereby the encoder and the decoder respectively model the E-step and M-step (Ghojogh et al., 2021).

Similarly to other nonlinear techniques, autoencoders exploit the idea that data are concentrated in a low-dimensional nonlinear manifold (Goodfellow et al., 2016). Thus, the encoder function constructs a mapping that is not sensitive to variations that are orthogonal to the manifold and is hence only sensitive to changes along the manifold directions. This makes autoencoders a parametric dimension reduction technique, more suitable for online inference and supervised tasks like classification and regression. This is an important advantage of autoencoders over all the above-mentioned non-linear techniques such as t-SNE and UMAP that do not learn a mapping of the manifold and cannot be applied to unseen data. This has led to several attempts to combine autoencoders and local structure-preserving graph-based techniques. For instance, Sainburg et al. (2021) proposed a parametric version of UMAP that replaces the gradient descent optimization stage with an autoencoder, thus obtaining comparable performance results. In a similar vein, Jia et al. (2015) proposed Laplacian autoencoders that introduce a locality-preserving regularization into the reconstruction loss of deep autoencoders. Another advantage offered by using autoencoder architectures is that they can be enhanced to fit different data types such as images using convolution layers (Chen et al., 2017b) and audio data using recurrent layers (Fabius et al., 2015).

For more details and examples the reader is referred to Bank et al. (2020) for a recent survey about autoencoder approaches and to Espadoto et al. (2021); van der Maaten et al. (2009) for a holistic overview of dimension reduction techniques, including linear and nonlinear methods.

1.3.4 Combined Dimension Reduction and Clustering

Beside visualization, dimension reduction is commonly used as a pre-processing step either to reduce computation and storage resources or to translate the original data into a space that is more propitious for the sought task. Data clustering is one of the tasks that most usually rely on dimension reduction (Drineas et al., 2004; Niu et al., 2011; Xu et al., 2003).

Non-negatice Matrix Factorisation and clustering

The Non-negative-Matrix Factorization (NMF) due to Lee and Seung (1999) belongs to the larger family of dimensionality reduction techniques. NMF attempts to decompose a positive data matrix into two non-negative factor matrices, whose product provides a good approximation to the original data.

In the context of text data, NMF seeks a decomposition of a document-word matrix $\mathbf{X}$ into two low dimensional factor matrices $\mathbf{Z} = (z_{ik}) \in \mathbb{R}_+^{n \times g}$ and $\mathbf{W} = (w_{jk}) \in \mathbb{R}_+^{d \times g}$, such

that $\mathbf{X} \approx \mathbf{Z}\mathbf{W}^\top$. To infer the latent factor matrices, NMF attemps to solve the following optimization problem:

$$\underset{\mathbf{Z},\mathbf{W}}{\text{minimize}}\ \Delta(\mathbf{X}, \mathbf{Z}\mathbf{W}^\top), \quad s.t.\ \mathbf{Z},\ \mathbf{W} \geq 0. \tag{1.5}$$

where Δ is a cost function that allows us to quantify the quality of the approximation of $\mathbf{X}$ by $\mathbf{Z}\mathbf{W}^\top$; Δ can be, for instance, the Frobenius norm or I-divergence. As NMF has an inherent clustering property, the document factor matrix $\mathbf{Z}$ is usually considered as a soft cluster membership matrix, where z_{ik} denotes the degree to which document i belongs to cluster k. A partition of the set of documents can then be obtained by assigning each document to the most likely cluster. Thus, NMF has become an essential unsupervised learning technique to analyze positive matrices arising in various areas such as pattern recognition, recommender systems, bio-informatics and text mining. Note that NMF can also be seen as a dimensionality reduction technique that seeks to decompose a positive data matrix into two lower dimensional latent factor matrices, restricted to be non-negative, whose product provides a good approximation to the original data. The first latent matrix contains the low dimensional representation of each row while the second contains the low dimensional representation of each column.

Although clustering is not the primary purpose of NMF, the latter has received a lot of interest in the clustering community resulting in a new class of clustering algorithms—based on NMF. In (Ailem et al., 2017b; Febrissy et al., 2022) the authors described Semantic-NMF, a novel non-negative factorization model which explicitly accounts for the semantic relationships among words. Similar to neural word embedding techniques, the model follows the distributional hypothesis so as to leverage the relationships between words. Formally, Semantic-NMF jointly decomposes the document-word and PPMI word-context matrices, with shared word factors. The intuition behind this approach is to map words having similar meanings roughly to the same direction in the latent space. More interestingly, by capturing more semantics, the model implicitly brings the embeddings of documents which are about the same topic closer to each other. This results in document factors that are even better for clustering. Moreover, they identify in which situations Semantic-NMF does provide the most significant improvements, which allows to gain further insights into the benefits of leveraging the word relationships.

Note that an extension of Semantic-NMF is also possible by exploiting the extension of NMF to NMTF (Non-negative Matrix Tri-Factorisation) as proposed by Salah et al. (2018).

Matrix factorisation approaches

Combining dimensionality reduction and clustering can be performed in several ways: in a two-phased manner usually called *tandem* approach, or simultaneously in an end-to-end fashion. Tandem approaches are made of two disjoint steps. First, the original data are embedded into a latent space of reduced dimension using any dimension reduction technique such as PCA for numerical data (Yau et al., 2016) and FAMD (Factor Analysis of Mixed Data) (Markos et al., 2020; van de Velden et al., 2019). The embedding step has proven effective in improving the clustering performance as shown in (Boutsidis et al., 2014; McConville et al., 2021). However, several authors have questioned the tandem approach, arguing that techniques exclusively made for dimension reduction are not sufficient to grasp the separability of underlying clusters and hence do not provide an optimal subspace for data clustering. For instance, Markos et al. (2019); van Aken et al. (2019); Vichi and Kiers (2001) stated that the reduced space provided by linear dimension reduction techniques like PCA may discard variables that help discriminate the underlying groups as long as they do not account for a sufficient amount of variation in the data; see for instance Chang (1983) for an illustration of this phenomenon. With this in mind, several approaches attempt to incorporate a clustering side into dimension reduction by optimizing both objectives at once. Reduced k-means (De Soete and Carroll, 1994) combines PCA and k-means within the same criterion, expressed as:

$$\underset{\mathbf{W},\mathbf{F},\mathbf{Z}}{\text{minimize}} \ \|\mathbf{X} - \mathbf{Z}\mathbf{F}\mathbf{W}^\top\|_F^2; \ \text{with} \ \ \mathbf{W}^\top\mathbf{W} = I_{d'}$$

$\mathbf{Z} = (z_{ik})_{n\times g}$ being the binary assignment (or membership) matrix where $z_{ik} = 1$ only if the ith data sample belongs to the kth cluster. $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_g]^\top$ is the centroid matrix of size $g \times d'$ where $\mathbf{f}_k$ is the centroid of the kth cluster in the reduced space. $\mathbf{W}$ is the orthogonal loading matrix of size $d \times d'$. $\mathbf{A}$, $\mathbf{F}$ and $\mathbf{Z}$ are first initialized and then updated by alternating the following steps until convergence:

1. Update the loading matrix with $\mathbf{A} = \mathbf{V}_{d'}\mathbf{U}_{d'}^\top$, where $\mathbf{U}_{d'}\mathbf{\Sigma}_{d'}\mathbf{V}_{d'}^\top$ is the truncated singular value decomposition of $(\mathbf{Z}\mathbf{F})^\top\mathbf{X}$.

2. Assign each data sample i to the cluster with the closest centroid in the sense of the k-means criterion: $u_{ik} = 1$ only if $k = \text{argmin}_h(\|\mathbf{A}^\top\mathbf{x}_i - \mathbf{f}_h\|^2)$.

3. Update the centroids with $\mathbf{F} = (\mathbf{Z}^\top\mathbf{Z})^{-1}\mathbf{Z}^\top\mathbf{X}\mathbf{A}$.

In the same vein, Factorial k-means (Vichi and Kiers, 2001) and Subspace k-means (Timmerman et al., 2013) aim at improving k-means clustering by finding a latent space that

best separates the data into groups following the k-means paradigm of inertia maximization. Such methods have shown their effectiveness, especially on synthetic data on which the tandem approach is likely to fail to discover the underlying groups. The reader can also refer to other close approaches such as (Allab et al., 2016, 2018). Probabilistic variants have also been proposed such as probabilistic principal component analyzer (Tipping and Bishop, 1999) and mixtures of factor analyzers (Ghahramani et al., 1996; McLachlan et al., 2003) that both use a variant of the Expectation Maximization algorithm to solve the log-likelihood maximization problem in a low-dimensional factor space. Recently, in (Labiod and Nadif, 2021), the authors proposed an effective clustering algorithm based on a matrix decomposition technique for learning a spectral data embedding, a cluster membership matrix, and a rotation matrix that closely maps out the continuous spectral embedding. Specifically, from a normalized similarity matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$, the proposed method, referred to as Regularized Dpectral Data Embedding (RSDE), can be seen as a procedure for finding the subspace that is the most informative about the clustering structure in the data. It is defined as the minimizing problem

$$\min_{\mathbf{B},\mathbf{M},\mathbf{Q},\mathbf{Z}} \left\|\mathbf{W} - \mathbf{B}\mathbf{M}^\top\right\|^2 + \lambda\|\mathbf{B} - \mathbf{Z}\mathbf{Q}\|^2 \quad \mathbf{B}^\top\mathbf{B} = \mathbf{I}, \mathbf{Q}^\top\mathbf{Q} = \mathbf{I}, \mathbf{Z} \in \{0,1\}^{n\times g}$$

where $\mathbf{Q} \in \mathbb{R}^{g\times g}$ is a rotation matrix that satisfies the orthogonality conditions $\mathbf{Q}^\top\mathbf{Q} = \mathbf{Q}\mathbf{Q}^\top = \mathbf{I}$. The nonnegative matrix $\mathbf{Z} = (z_{ik})$ of size $(n \times g)$ is a cluster membership matrix, $\mathbf{B} = (b_{ij})$ of size $(n \times g)$ is the embedding matrix, and $\mathbf{Q} = (q_{ij})$ of size $(g \times g)$ is the orthonormal rotation matrix that most closely maps $\mathbf{B}$ to $\mathbf{Z} \in \{0,1\}^{n\times g}$. $\mathbf{M} \in \mathbb{R}^{n\times g}$ is an auxiliary variable introduced to improve the efficiency of the optimization.

Those methods are generally used on high-dimensional numerical data like gene expression microarrays (McLachlan et al., 2003) and RNA-sequencing data (Sun et al., 2019). For categorical data, usually represented by binary dummy variables, the criterion of the Multiple Correspondence Analysis (MCA) is often used with the k-means objective either sequentially (Arimond and Elfessi, 2001) or simultaneously (Hwang et al., 2006; van Buuren and Heiser, 1989) with prevalent applications to social media profiling (van Dam and van de Velden, 2015) and preference data analysis (Takagishi et al., 2019).

Autoencoders have also often been used for clustering purposes as part of a tandem approach where they are followed by a clustering algorithm such as k-means (Huang et al., 2014; Tao et al., 2021; Tian et al., 2014). Autoencoders are also widely used as the embedding module in simultaneous approaches combining dimension reduction and clustering into the same neural network. In particular, variational autoencoders have been extensively

used for data clustering, generally with a Gaussian prior (Dilokthanakul et al., 2016; Yang et al., 2019a), sometimes in a two-stage (tandem) manner (Lim et al., 2020).

Deep clustering approaches

With the advent of Deep Neural Networks (Schmidhuber, 2015) and the multiplication of use-cases involving either unstructured or high-dimensional data, substantial energy has been devoted to addressing data clustering using deep architectures. Deep clustering approaches generally include a representation learning component based on autoencoders (cf. Section 1.3.3) that first translates the input data into a latent space. Beyond reducing the dimensionality, the aim of learning a latent representation is to project the input data into a "clustering-friendly" embedding space (Yang et al., 2017), supposed to better separate the underlying groups. After Xie et al. (2016) have proposed Deep Embedding Clustering (DEC) based on two separate modules: one for data embedding and one for clustering, Guo et al. (2017) proposed a variant called Improved Deep Embedding Clustering (IDEC) where the optimization of both modules is performed simultaneously through a joint loss function similar to the objectives optimized by DEC. In the same vein, Deep Clustering Network (DCN) proposed in (Yang et al., 2017) jointly optimizes, in an alternate fashion, a latent embedding module (autoencoder) and a clustering component with a k-means loss function. Recently, Fard et al. (2020) proposed Deep k-means (DKM) which, like DCN, optimizes a k-means objective function along with the reconstruction error of an autoencoder, simultaneously rather than alternately. Both DCN and DKM perform a pre-training phase of the embedding module and use k-means to initialize the cluster centroids (applied to the pre-trained latent representation). The difference between two-phased and simultaneous approaches is depicted in Figure 1.3.

Deep neural networks have also been used in community detection, where stacked autoencoders (Bengio et al., 2006; Vincent et al., 2010) are particularly popular. Stacked autoencoders consist in learning more than one shallow autoencoder in such a way that the ℓth autoencoder provides a latent representation that in turn feeds the autoencoder $\ell + 1$. The autoencoders are trained separately and stacked together after the training process has been completed. For instance, Yang et al. (2016) utilized stacked autoencoders combined with the k-means algorithm to maximize the modularity for graph-clustering. Another recent approach makes use of stacked encoders and combines four different similarity matrices under an ensemble clustering framework for community detection (Xu et al., 2020). Graph Convolutional Networks (GCN) (Kipf and Welling, 2017) are also often used to tackle graph-clustering tasks. Graph convolution consists in successively applying learned filters at each layer, thus generalizing the concept of Convolutional Neural Networks (LeCun

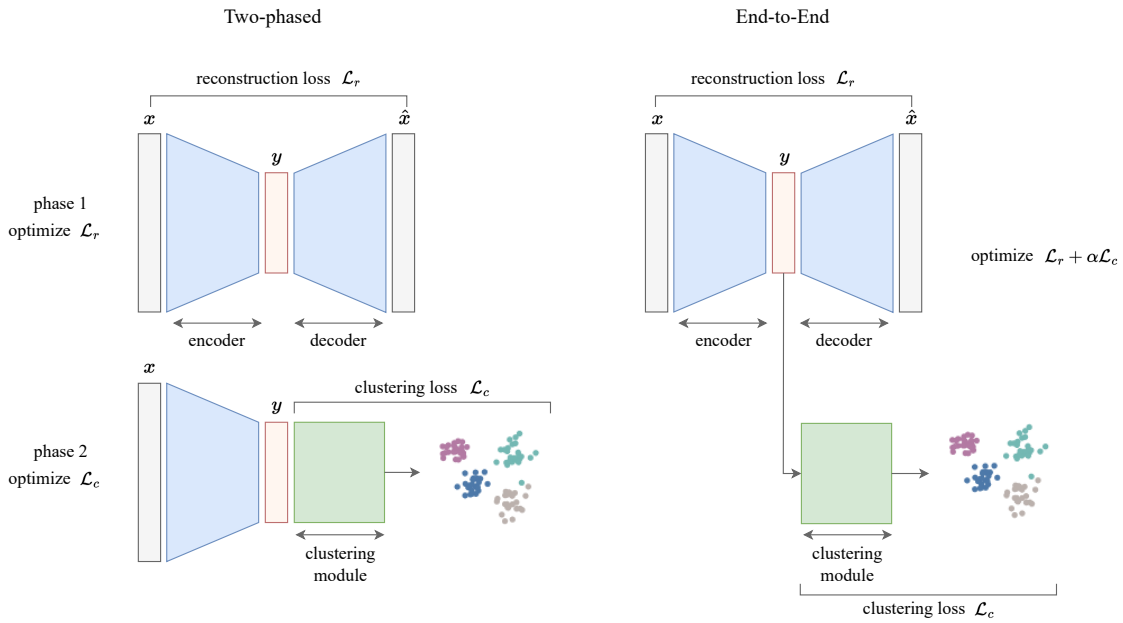

Figure 1.3: Difference between two-phased and end-to-end (simultaneous) deep clustering approaches.

et al., 1995) to graphs. The aim of GCNs is to find representations of the nodes based on both the graph structure and the nodes' attributes. The obtained representations can be used for several supervised, semi-supervised and unsupervised tasks. For instance, Bo et al. (2020) included a GCN module combined with autoencoders to train an end-to-end graph clustering model.

Many other classical clustering approaches have inspired deep neural approaches, e.g. model-based clustering (Ben-Yosef and Weinshall, 2018; Tian et al., 2019), and various neural architectures have been leveraged to perform clustering like Generative Adversarial Networks (Jia et al., 2019) and Recurrent Neural Networks (Lin et al., 2019). More details and examples are provided regarding deep clustering approaches in (Aljalbout et al., 2018; Zhou et al., 2022).

1.3.5 Applications

Linear and nonlinear dimension reduction can be used to achieve diverse objectives such as data compression, visualization and denoising. Additionally, dimension reduction techniques are often used to perform feature extraction in order to boost other machine learning tasks such as speech recognition (Tang and Rose, 2008), recommender systems (Sarwar et al., 2000) and clustering as seen in Section 1.3.4. Inversely, the capacity of clustering to synthesize data can serve dimension reduction purposes. For instance, in (Karami, 2019),

fuzzy clustering is used as a dimension reduction technique on text data and has proven to be faster and more effective than linear dimension reduction on the text classification task.

Like clustering, dimension reduction techniques are very popular in the field of molecular biology either as a pre-processing or a visualization tool (Meng et al., 2016). This is explained by the fact that gene expression data are often described by thousands of features, among which only a handful might be significant. Capturing useful information from these data allows for detecting malignant diseases (e.g. cancer) and can be a challenging task. In particular, manifold techniques have widely been used on high-dimensional gene expression data such as UMAP (Becht et al., 2018), t-SNE (Khan et al., 2018) and multidimensional scaling (Alexandrov et al., 2014).

Dimension reduction is also widely used on text data for various objectives such as clustering (Kadhim et al., 2014), visualization (Schubert et al., 2017) and classification (Uysal, 2018), with application in several fields like health care (Halpern et al., 2012), education (Kakkonen et al., 2008) and budget management (Williams and Gong, 2014).

1.4 Anomaly Detection

Anomaly detection, sometimes called outlier detection, is an unsupervised discipline which seeks to identify anomalous samples, also called outliers, in a dataset. An outlier (an anomaly, an exception) is defined by Hawkins (1980) as "an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism".

Given a dataset $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\},\ \mathbf{x}_i \in \mathbb{R}^d$, we define the objective of an anomaly detection system as learning a mapping function $f(.) : \mathcal{D} \mapsto \mathbb{R}$ that allows for distinguishing outliers from the global behavior of data samples. To address this objective, a wide range of methods exist in the literature, based on a variety of approaches. In this section, a brief description is provided for four main families of methods namely distance-based, density-based, reconstruction-based and clustering-based. However, other categories of methods exist such as one-class classification (Hempstalk et al., 2008; Schölkopf et al., 2001) and subspace-based approaches (Keller et al., 2012; Müller et al., 2012). For a more detailed overview of the existing approaches for anomaly detection, the reader is referred to Chandola et al. (2009); Hodge and Austin (2004) and Wang et al. (2019).

It is worth noting that the majority of anomaly detection approaches suppose that the training set is free of outliers and contains solely normal samples (inliers). This principle comes from the one-class classification scheme (Schölkopf et al., 1999) that aims at detecting *novelty* by discovering instances in the test set that do not resemble those of the training

set. Novelty detection can be very useful in medical research and real-time monitoring systems such as intrusion detection systems where already-known anomalies are discarded. However, the use of inlier-only training sets is also commonplace in the outlier detection community. Technically speaking, this should not be a major obstacle since it is always possible to bypass the inlier-only restriction and allow the training set to be contaminated with potential anomalies (Ding et al., 2022a; Manolache et al., 2021). It is hence important to assess the robustness of such approaches in the presence of outliers in the training set, which most of anomaly detection studies do not provide for. Most importantly, although the inlier-only scheme may reasonably be considered semi-supervised (Aggarwal, 2017), it is also used in supposedly unsupervised approaches (Akcay et al., 2019; Qiu et al., 2021; Ruff et al., 2018, 2019; Schlegl et al., 2019) without studying the effect of outlier contamination, which is a key determinant of the robustness of any unsupervised approach. Accordingly, the dividing line between anomaly detection and novelty detection can be somewhat fuzzy, each one borrowing elements from the other, without always setting a realistic framework. For instance, OC-SVM (Schölkopf et al., 2001) and Isolation Forest (Liu et al., 2008) are well-known techniques that are used for both anomaly and novelty detection, depending on the application purpose as well as the experimental setup.

1.4.1 Distance-based and Density-based Approaches

One of the most classical families of methods is distance-based outlier detection (Knorr and Ng, 1998). Approaches in this category are based on the assumption that anomalies are isolated from the rest of the data instances in terms of a predetermined distance measure. In (Ramaswamy et al., 2000) outliers are defined as the data samples whose distance to their kth nearest neighbor is highest (maximum distance), whereas in (Angiulli and Pizzuti, 2002) and Eskin et al. (2002), the average distance with the kth nearest neighbors is used. One of the drawbacks of distance-based methods is that they do not work on non-homogeneous data that can be structured into several underlying groups.

On the other hand, density-based approaches consider instances lying in low-density regions as outliers. For instance, Local Outlier Factor (LOF) (Breunig et al., 2000), the best-known density-based outlier detection method, computes for each data sample the ratio between the average local density of its k nearest neighbors and its own local density. This allows handling datasets with various density regions since anomalous samples are considered as data points with a density that is significantly lower than the density of its nearest neighbors.

1.4.2 Reconstruction-based Approaches

As mentioned previously, dimension reduction techniques are commonly used for denoising various types of data by reconstructing a cleaner version of the input data. This capability is also often harnessed in the context of anomaly detection, with the assumption that a well-generalizing model would be unable to accurately reconstruct atypical samples. PCA is naturally well suited for this purpose since it incorporates the process of compression and reconstruction (Huang and Kasiviswanathan, 2015; Jablonski et al., 2015). Once the linear mapping $\mathbf{W}_{d'}$ is computed (cf. Section 1.3.2), the reconstruction error of the data point $\mathbf{x}_i$ can be expressed as $\|\mathbf{x}_i - \mathbf{W}_{d'}\mathbf{W}_{d'}^{\top}\mathbf{x}_i\|$, which would constitute its anomaly score.

In order to overcome the limitations of PCA, nonlinear methods are used such as autoencoders, using the reconstruction error of each data sample (Equation 1.4) as the anomaly score (Schreyer et al., 2017). In (Chen et al., 2017a), an ensemble of independently trained autoencoders is proposed whereby the median of the m reconstruction error values is used as the final anomaly score. Besides, in each autoencoder, random connections are removed in order to augment their diversity. Another way to address the diversity problem would be to use different bottleneck architectures in order to ensure that each autoencoder is trained with a different level of capacity (cf. Section 1.3.3).

1.4.3 Clustering-based Anomaly Detection

Anomaly detection can also be achieved through clustering approaches, exploiting the concept of adherence to underlying groups. One way of identifying outliers is relying on clustering methods that do not force every data sample to belong to a cluster, hence considering the isolated data points as outliers. Examples of such approaches are DBSCAN (Ester et al., 1996), CURE (Guha et al., 1998) and WaveCluster (Sheikholeslami et al., 1998). Other approaches use clustering algorithms and consider the samples that belong to small and/or low-density clusters as outliers (Eskin et al., 2002; Jiang et al., 2001). Note that these approaches do not necessarily provide an anomaly score in $\mathbb{R}$ but rather a binary label indicating whether or not a given data point is considered anomalous. This can make it difficult to compare or rank anomalies based on their relative abnormality.

Another way of identifying anomalies in a given dataset is to make use of a membership score that determines, within each cluster, the samples that are most likely to constitute outliers. The way of computing the score naturally depends on the clustering algorithm used and can be, for instance, the distance to the centroid in the case of k-means and fuzzy k-means. In the same spirit, model-based clustering methods can also be used to identify

outliers with various priors such as Gaussian (Mahadevan et al., 2010), Dirichlet (Fan et al., 2011) and von Mises-Fisher (Zhuang et al., 2017).

1.4.4 Applications

Unsupervised anomaly detection finds application in diverse domains such as health insurance (Joudaki et al., 2015), banking (Ahmed et al., 2016) and cybersecurity (Gogoi et al., 2011). Anomalies are generally perceived as negative phenomena like fraud, intrusions and health issues. However, finding anomalies in certain domains can induce valuable advances. This is the case in the astrophysics field where spotting anomalies in data may lead to important scientific breakthroughs (Rebbapragada et al., 2009). Astrophysics and astronomy are, indeed, good examples of research domains involving massive volumes of unlabeled data to analyze. Another discipline that generates large volumes of data is the climate science domain, in which an anomaly may consist, for example, of an unusual weather event (Das and Parthasarathy, 2009).

Furthermore, anomaly detection is often applied to images and videos such as optical tomography images (Schlegl et al., 2019) and surveillance videos (Xiang and Gong, 2005). Use-cases involving text data include spam detection (Kumar et al., 2019) and insider threat detection (Kim et al., 2019) and aim at spotting atypical phrase structures or suspicious vocabulary usage. Another type of data that usually requires monitoring and outlier detection is log data. For instance, Kuna et al. (2014) proposed to combine the results of several algorithms such as LOF and DBSCAN in order to minimize the risk of false alarms in audit logs analysis. Besides the data type, the formatting of data may also differ. For instance, three-way tensors can be used to characterize each data sample with a fixed sized matrix that may represent pair-wise interactions. Tensor decomposition methods are then used on such representations in order to detect, for instance, cyber attacks (Eren et al., 2022) and botnets (Kanehara et al., 2019).

Anomaly detection is also widely used in graph data with different purposes. Most usually, anomalous *nodes* are sought with typical application in social media (Heard et al., 2010) and the detection of fake users (Hooi et al., 2016; Wang et al., 2011). Other use-cases seek to identify anomalous *edges* or connection as in (Chang et al., 2021), where streaming anomalous connections are identified based on the likelihood of the observed frequency of each edge. On the other hand, some graph-based approaches focus on spotting suspicious *groups*, usually using clustering techniques. For instance, Yu et al. (2015b) discovered anomalous groups of nodes in social media by analyzing the collective behaviors of the users. More methods and applications for real-world graph data can be found in (Akoglu et al., 2015).

1.5 A Recurring Issue in Deep Unsupervised Learning

In the recent years, deep neural networks have significantly gained in popularity in the unsupervised realm of machine learning, leading to a series of breakthroughs in several research domains like clustering, information retrieval and anomaly detection. However, it is essential to put those approaches in the unsupervised context where they belong. Indeed, deep learning approaches are known for being highly sensitive to hyperparameter settings (Hutter et al., 2019; Ozaki et al., 2017) and for having several hyperparameters to tune, including categorical (e.g. the choice of the activation function) and numerical (e.g. the weight decay). Some of those hyperparameters (e.g. the learning rate) are crucial, in the sense that slight changes in the chosen value can lead to serious performance drops (Ozaki et al., 2017). Beside those hyperparameters, additional ones are usually used for a specific optimization purpose such as regularization coefficients (Bo et al., 2020; Fard et al., 2020). Yet another important choice to make concerns the architecture used (e.g. the type of layers and the number of neurons per layer), which can be decisive for both the quality of the model and its computational complexity. In a supervised setting where a performance metric can realistically be optimized, this is only partly inconvenient since it considerably increases the computational cost of already onerous training models but remains feasible. This is acheived via hyperparameter tuning, which consists in finding the combination of hyperparmeter values that maximizes a certain criterion (usually the performance of the model). Depending on the number of hyperparameters and the number of possible values of each hyperparameter, several approaches may be used in order to address the tuning problem. For instance, the grid-search approach is an exact algorithm that evaluates the whole set of combinations and returns the one yielding the best performance score. However, the problem can rapidly become intractable given that the number of possible combinations grows exponentially with the number of hyperparameters (Yang and Shami, 2020). In such case, near-optimal solutions are sought using heuristics like random search (Bergstra and Bengio, 2012) and particle swarm optimization (Lorenzo et al., 2017).

In a real-world unsupervised scenario, though, it is not only inconvenient but impractical to perform hyperparameter tuning since labels are only available in research, for evaluation purposes. Despite that, it is common to encounter in the research community supposedly unsupervised works that perform different levels of supervised hyperparameter tuning as shown in Table 1.1. We define "supervised tuning" as the process of setting different values of hyperparameters for each dataset based on external information (labels) not supposed to be available in real life. This includes automatic tuning carried out on a validation set (Fard et al., 2020; Munir et al., 2018) as well as arbitrary choices made for each dataset (Ji et al.; Yang et al., 2016). Furthermore, as mentioned earlier, we distinguish

Table 1.1: Examples of unsupervised approaches performing supervised hyper-parameter tuning (yes is bad, no is good). In the "Task" column, "clustering" stands for general-purpose clustering, usually applied to images and text and tabular data. The last column is relative to indications that would help the user choose the tuned hyperparameters and/or model design on a new unlabeled dataset.

Method	Task	Supervised parameter tuning	Supervised architect. tuning	No sensitivity analysis	No indication given
AE-k-means (Huang et al., 2014)	image clustering	yes	yes	yes	yes
SAE (Tian et al., 2014)	graph clustering	yes	yes	yes	yes
DEC (Xie et al., 2016)	clustering	yes	no	no	no
DNR (Yang et al., 2016)	graph clustering	unk[2]	yes	yes	yes
DCN (Yang et al., 2017)	clustering	yes	yes	no[1]	yes
IDEC (Guo et al., 2017)	clustering	yes	no	no[1]	yes[3]
DeepCluster (Tian et al., 2017)	clustering	yes	no	no	yes
DSC-Nets (Ji et al.)	clustering	yes	yes	yes	yes
DeepAnT (Munir et al., 2018)	anomaly detection	yes	no	no	yes
DAGMM (Zong et al., 2018)	anomaly detection	yes[4]	yes	no	yes
DKM (Fard et al., 2020)	clustering	yes	no	yes	yes
NeuTraL (Qiu et al., 2021)	anomaly detection	yes	no[5]	no[6]	yes

[1] Sensitivity analysis provided for one dataset.
[2] Unknown. No details provided about the chosen values of the hyperparameters.
[3] Indication given regarding only one of the six hyperparameters.
[4] The batch size and the number of epochs are set arbitrarily for each dataset.
[5] Note however that different numbers of transformations are used which influences the num. of neurons.
[6] Sensitivity analysis provided for two datasets.

optimization hyperparameters and model design (or architecture) hyperparameters. We can observe that most of the mentioned approaches perform supervised hyperparameter tuning (first column of the table) and do not provide indications as to how to choose the hyperparameters when no labels are available (last column of the table). In addition, half of the approaches choose different architectures of the network from one dataset to another without necessarily explaining these choices. In (Yang et al., 2016), it is not possible to say how the hyperparameter choices have been made since no information is provided about the values used by the authors. This raises a reproducibility issue (Haibe-Kains et al., 2020), which is related to our subject but beyond the scope of this discussion. Table 1.1 contains only a few examples, but supervised tuning is unfortunately common practice in the unsupervised learning domain and many other examples could have been included (Park et al., 2020; Peng et al., 2019; Yang et al., 2019a).

This topic has all too seldom been discussed in the literature and is certainly worth a deeper investigation. Nonetheless, the issue may be addressed by relying on either (i) unsupervised hyperparameter tuning (Purohit et al., 2020) based on internal rather than external criteria, (ii) meta-learning that allows for generalizing on new unlabeled datasets using labeled data (Zhao and Akoglu, 2022; Zhao et al., 2020), or (iii) ensemble approaches, known for their robustness (Affeldt et al., 2020b; Ding et al., 2022a; Vega-Pons and Ruiz-Shulcloper, 2011). For instance, to overcome the hyperparameter tuning problem and still benefit from the capability of deep autoencoders to discover nonlinear complex structures in data, Affeldt et al. (2022) proposed to use networks with different hyperparameter settings and combine them as part of an ensemble approach for spectral clustering. To this end, m affinity matrices, computed via the latent representations of m autoencoders, are concatenated and fed into a spectral clustering component which provides the final consensus clustering partition. In the context of outlier detection, Ding et al. (2022a) provided an empirical sensitivity study of several well-known approaches, showing that model selection is inevitable. They also proposed ROBOD, an ensemble approach that combines (average) anomaly scores provided by the same model with different combinations of hyperparameter values. The proposed framework is based on BatchEnsemble (Wen et al., 2020) which allows for training the same neural network with different parameters in a more frugal way, using a shared matrix of weights instead of training each model separately. ROBOD substantially improves the robustness of existing deep anomaly detection models and offers a good alternative for hyperparameter tuning. However, although the authors provided for the presence of models of different depths by multiplying the shared matrix by a zero-mask, it is unclear how models with very different architectures can be integrated, e.g. uni/bi-directional, self-attention/RNN, etc.

It is worth noting that the supervised tuning phenomenon is not confined to deep approaches and that the other unsupervised techniques are not necessarily spared. Only, as mentioned earlier, deep neural networks are much more prone to hyperparameter sensitivity and involve substantially more parameters to tune. Also, we do not discuss the open problem of setting the number of groups in clustering approaches, even though it can utterly be viewed as a hyperparameter as it can significantly affect the clustering quality (Xu and Gong, 2004; Xu et al., 2003).

1.6 Conclusion

In this chapter we introduced a summary of the current state of research is four major unsupervised learning fields: text representation, clustering, dimension reduction and anomaly detection; including some recent advances and developments as well as real-world data applications. Some of these disciplines are related to each other, if not complementary. Indeed, we show in the next chapters how different tasks may be combined to achieve a given unsupervised objective, such as text representation, dimension reduction and clustering. Naturally, this overview is by no means exhaustive, but it provides a general sense of the state of the art.

We also highlighted an important issue often encountered in the unsupervised learning field, especially in deep neural approaches, which is the supervised hyperparameter tuning. In the remainder of this thesis, a special effort is made to address this issue in all of our contributions in the spirit of reusability, robustness, and transparency.

Chapter 2

Unsupervised Methods for the Study of Transformer Embeddings

Over the last decade neural word embeddings have become a cornerstone of many important text mining applications such as text classification, sentiment analysis, named entity recognition, question answering systems, etc. Particularly, Transformer-based contextual word embeddings have gained much attention with several works trying to understanding how such models work, through the use of supervised probing tasks, and usually emphasizing on BERT. In this chapter, we propose a fully unsupervised manner to analyze Transformer-based embedding models in their bare state with no fine-tuning. We more precisely focus on characterizing and identifying groups of Transformer layers across 6 different Transformer models.

2.1 Introduction

Transformer-based word embeddings provided by neural language models are today increasingly used as the initial input to many text mining applications where they greatly contribute to achieve impressing performance levels. This has motivated a growing number of researchers to investigate the reasons behind this effectiveness as part of the general effort to unlock the black box of AI models. Since a Transformer model produces several embeddings for each word (one for each layer of its deep architecture), it is natural to study the nature of the embeddings learned at the different layers of the model. So far, the common way of doing this is to feed them as input to some supervised probing tasks (text classification, question answering, etc.) and then measure how well they perform on these tasks. From the observed performance, and depending on the probing task used, one may deduce, for example, that a given set of layers seems to be good at capturing some

features of language while another set seems to encode another kind of information. While these experiments have allowed to draw some conclusions, the observed results depend both on the tasks and the train and test datasets, and so are not always generalizable. This observation prompted us to explore if it could be possible to gain additional insight into the behavior of the layers without having to rely on supervised probing tasks and external datasets.

In this chapter, we propose unsupervised techniques that completely dispense of probing tasks, and demonstrate their interest by applying them to real datasets and several widely used Transformer models. The contributions of the study are as follows:

1. We propose a set of unsupervised methods that allow to gain insights into the nature of the embeddings available at the different layers of a Transformer model, and how these embedding layers relate to each other. This approach, which directly leverages the intrinsic features of the layers, is in contrast to other studies that rely on probing tasks.

2. The experimental section shows that applying these methods on real datasets allows to acquire new knowledge about the layers of several Transformer models that seem to best perform on the important word clustering task.

3. Also, while most layer interpretation studies have so far focused mainly on BERT we provide a performance comparison for 3 different models namely BERT, RoBERTa and ALBERT, in both their base and large versions.

2.2 Related Work

In the supervised learning realm, a growing body of research has been devoted to investigating the linguistic features learned by contextual word embedding models including LSTM-based models as in (Peters et al., 2018a) and Transformer-based models like BERT as in (Tenney et al., 2019a). Both authors agreed to say that early layers encode most local syntactic phenomena while more complex semantics appear at the higher layers. In (Liu et al., 2019a), the authors evaluated the performance of contextualized word representations on several supervised tasks and compare layers with each other, including ELMo, BERT (base and large) and OpenAI Transformer models. They especially observe that Transformers' middle layers allow for a better transferability. On the other hand, the authors in (Hao et al., 2019) observed that the early layers of BERT are more invariant across tasks and hence more transferable. It has also been shown in (van Aken et al., 2019) that, after fine tuning BERT on Question Answering, the model acts in different phases starting from capturing

the semantic meaning of tokens in the first layers to separating the answer token from the others in the last layers. It has been concluded that the closer we get to the last layer, the more task specific the representations are. This explains the results obtained in (Kovaleva et al., 2019) which studies the changes between pre-trained and fine-tuned BERT-base model in terms of attention weights. A significant change in the two last layers in terms of cosine similarity between original and fine-tuned attention weights has been observed on 6 GLUE tasks. The authors deduced that the BERT-base's two last layers learn more task specific features. Several papers focus on identifying the linguistic structure implicitly learned by the models (Clark et al., 2019; Jawahar et al., 2019). For example, Goldberg (Goldberg, 2019) evaluates how well BERT captures syntactic information for subject-verb agreement. Ethayarajh (2019) tried to assess how context-specific are the representations at the different layers of ELMo, BERT and GPT-2. More examples of works dedicated to understanding neural language models are given in (Belinkov and Glass, 2019; Rogers et al., 2020).

In contrast to the above studies we propose to identify coherent groupings of layers, based on the intrinsic characteristics of the layers and not only by resorting to external probing tasks.

2.3 Unsupervised Methods for Layer Analysis

Deep Transformer models may have dozens of layers (see Table 2.3). In order to better understand their behavior we argue that it is useful to compare them, and try to identify groups with similar characteristics.

2.3.1 Matrix and Vector Representation of Layers

In this section, we propose several alternative (matrix- and vector-based) representations for a Transformer layer, thus allowing to study their correlations from multiple points of view. Given a dataset of n words, and a Transformer model with b layers and embedding dimension d, the dataset can be represented by b different matrices $\mathbf{X}_1, \ldots, \mathbf{X}_b$ of size $n \times d$, where each matrix $\mathbf{X}_\ell$ corresponds to the dataset at the ℓ-th layer. An alternative way of representing a layer ℓ is by averaging the rows of its $\mathbf{X}_\ell$ matrix, leading to a vector representation $\mathbf{v}_\ell$ of the layer. Additional intermediate data structures are then computed from these initial representations (Table 2.1). The pseudo-code in Algorithm 1 describes in detail how these data structures are created and used during the analysis process.

Table 2.1: Definitions and notations

Symbol	Description
n	Number of words of the dataset.
d	Number of dimensions: 768 for base models and 1024 for large ones.
b	Number of layers: 12 for base models and 24 for large ones.
$\mathbf{X}_\ell$	Matrix of size $(n \times d)$: data matrix of layer ℓ (cf. Figure 2.1).
$\mathbf{x}_{\ell i}$	The ith row of $\mathbf{X}_\ell$.
$\mathbf{S}_\ell$	Matrix of size $(n \times n)$: corresponds to the square matrix of $\mathbf{X}_\ell$.
$\mathbf{v}_\ell$	Vector of size d: computed for a layer ℓ as the average of rows of $\mathbf{X}_\ell$.
$\mathbf{r}_\ell$	Vector of size n: similarity ranks of words regarding $\mathbf{v}_\ell$.

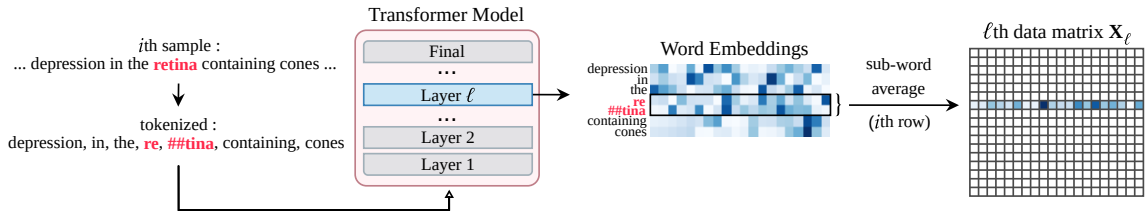

Figure 2.1: Construction of the data matrix $\mathbf{X}_\ell$ from the contextual word embeddings provided by the ℓth layer. The context of the word "retina" is used to compute its embedding, which constitutes the ith sample vector representation in the $\mathbf{X}_\ell$ data matrix w.r.t. to the layer ℓ.

2.3.2 Measuring the Correlations between Layers

In this section and the following one, we propose unsupervised methods for comparing the layers against each other the goal being to exhibit layers that share some characteristics.

When using a matrix representation for the layers (the $\mathbf{X}_\ell$ matrices of Table 2.1), an appropriate correlation measure is the R_v coefficient (Robert and Escoufier, 1976) which can be used in order to visualize the layers' similarities and distinguish any possible bloc structures. The R_v coefficient can be interpreted as a non centered squared coefficient of correlation between two given data matrices $\mathbf{X}_\ell$ and $\mathbf{X}_{\ell'}$ (cf. Algorithm 1). This proportion varies between 0 and 1 and the closer to 1 it is, the better is $\mathbf{X}_{\ell'}$ as a substitute for $\mathbf{X}_\ell$ (and *vice-versa*) to characterize the n samples of the dataset. In order to draw a similarity tendency across layers, we compute for each layer ℓ an *Average*-R_v which corresponds to the mean of R_v values between the layer ℓ and the rest of the layers. The heatmap representation of these values allows to spot groupings of layers.

The vector representation of the layers (the $\mathbf{v}_\ell$ vectors) allow for other possibilities. They can be used as input to a clustering algorithm (this will be described in Sections 2.3.3

Algorithm 1: Unsupervised Process of Layers' Analysis

Input: a dataset D of n words; a Transformer model M with b layers and embedding dimension d; a clustering algorithm $\mathcal{C}$; a ranking function $rank$.

Step 1. Build matrix and vector representations of layers, for each $\ell = 1 \dots b$:

$\mathbf{X}_\ell \leftarrow$ vertical stacking of the n word embeddings provided by the ℓth layer

$\mathbf{S}_\ell \leftarrow \mathbf{X}_\ell \mathbf{X}_\ell^T$

$\mathbf{v}_\ell \leftarrow \sum_i \mathbf{x}_{\ell i}$, where $\mathbf{x}_{\ell i}$ is the ith row of $\mathbf{X}_\ell$

$\mathbf{e}_{\ell i} \leftarrow$ cosine similarity between the word vectors $\mathbf{x}_{\ell i}$ and $\mathbf{v}_\ell$, $i = 1 \dots, n$

$\mathbf{r}_{\ell i} \leftarrow rank(\mathbf{e}_{\ell i})$, $i = 1 \dots, n$

Compute the R_v coefficient:

$$R_v(\mathbf{X}_\ell, \mathbf{X}_{\ell'}) = \frac{trace(\mathbf{S}_\ell \times \mathbf{S}_{\ell'})}{\sqrt{trace(\mathbf{S}_\ell^2) \times trace(\mathbf{S}_{\ell'}^2)}},\ \forall \ell, \ell' = 1, \dots, b$$

Compute the Spearman coefficient:

$$\rho(\mathbf{r}_\ell, \mathbf{r}_{\ell'}) = \frac{6 \sum_{i=1}^{n} (\mathbf{r}_{\ell i} - \mathbf{r}_{\ell' i})^2}{n(n^2 - 1)},\ \forall \ell, \ell' = 1, \dots, b$$

Step 2. Identify groups of layers

Visualize the R_v and Spearman coefficients as heatmap matrices.

$\mathbf{V} \leftarrow$ vertical stacking of the b vectors $\mathbf{v}_\ell$, $\ell = 1 \dots b$

$clusters \leftarrow \mathcal{C}(\mathbf{V})$

Visualise $clusters$

Step 3. Interpret the groups identified in step 2.

and 5.3). But they can also serve as a basis for measuring the correlations between layers. For each layer ℓ, we first compute the cosine similarity of its vector $\mathbf{v}_\ell$ with all the word vector $\mathbf{x}_{\ell i}$. A ranking vector $\mathbf{r}_\ell$ is then computed where $\mathbf{r}_{\ell i}$ is the ranking assigned to word i by layer ℓ. Since for two layers ℓ and ℓ' $\mathbf{r}_\ell$ and $\mathbf{r}_{\ell'}$ contain word ranks, a suitable measure of comparison is the Spearman correlation coefficient ρ that measures the rank correlation between two variables. From the ρ values between each pair of layers we can construct a heatmap matrix of size $b \times b$ which, as with the matrix of R_v values, also allows to identify groupings of layers (cf. Algorithm 1).

2.3.3 Clustering Layers

The next step of the study is to perform a cluster analysis to confirm the potential groups using the techniques described in the previous section. The data samples are the b layers of a given model where each layer ℓ is represented by its corresponding average vector $\mathbf{v}_\ell$. In theory, any kind of clustering algorithm could be used at this stage. In practice, since the number of layers is relatively low and the number of cluster is unknown (although the

techniques of the previous section can give an estimate of it), we often used Agglomerative Hierarchical Clustering (AHC) methods in our experiments. The hierarchical arrangement of the samples provided by the dendrograms indeed allows for a better interpretation of the clustering results as will be shown in the experiments section.

2.3.4 Interpreting Layers

In order to provide a more qualitative analysis of layers' behavior, we use the vector representation $\mathbf{r}_\ell$ and visualize the ranking of the first m words regarding their similarity to $\mathbf{v}_\ell$. We can also deepen our analysis of layers by using the interpretation abilities offered by dimension reduction techniques such as the Principal Component Analysis (PCA). When applying PCA on the $\mathbf{X}_\ell$ matrices, the samples are the word representations and the features represent the dimensions of the embeddings. The cos^2 measure denotes the correlation between a principal component and a given dimension (feature). It also measures the quality of representation of the feature, which allows us to select only the features that are more influential for interpretation.

2.4 Experiments

In this section, we first apply the process described in Algorithm 1 to several word datasets, in a step by step manner. Then, in order to validate the above methods and better understand the results they provide, we cluster our word datasets and evaluate each layer in terms of clustering performance. To achieve that, we perform word clustering experiments on the $\mathbf{X}_\ell$ matrices. Each clustering run provides a partition containing the cluster label of each word. To evaluate the partition obtained with each layer, we rely on a standard external measure for assessing clustering quality, namely Normalized Mutual Information (NMI) measure (Strehl and Ghosh, 2002).

2.4.1 Datasets and Models Used

The datasets of size n used in the experiments are described in Table 2.2. The first dataset, referred to as UFSAC3, is extracted from the UFSAC dataset (Vial et al., 2018) which consolidates multiple popular datasets annotated with WordNet (such as SemEval and SensEval) into a uniform format. The examples are manually divided into three topics: Body, Botany and Geography. The second dataset, UFSAC4, is a slightly more difficult dataset since it includes a fourth class (words related to Information Technology) and is augmented with some polysemous examples (such as "lobes" which appears in Body and

Botany with different contexts). The third dataset yahoo4, is extracted from the Yahoo! Answers dataset (Zhang et al., 2015) by manually selecting sets of words for each category and some corresponding contexts.

Table 2.3 describes the 6 pre-trained Transformer-based embedding models used for the experiments, without any fine-tuning.

Table 2.2: datasets description: k denotes the number of clusters.

datasets	n	k	clusters' sizes
UFSAC3	583	3	body: 266, geo: 227, botany: 90
UFSAC4	691	4	body: 275, geo: 227, botany: 99, IT:90
yahoo4	528	4	finance: 150, science-maths.: 152, computers-internet: 144, music: 82

Table 2.3: Transformer models' description.

	b	d	vocab size
BERT-base-cased	12	768	28,996
RoBERTa-base			50,265
ALBERT-base-v2			30,000
BERT-large-cased	24	1024	28,996
RoBERTa-large			50,265
ALBERT-large-v2			30,000

2.4.2 Investigating the Correlations between Layers

For comparing the layers with each other, we experimented with the R_v coefficient and the Spearman's rank correlation coefficient (cf. Section 2.3.2) . Figure 2.2 shows the similarities computed between the layers of each model in terms of the R_v coefficient which uses the matrix-based representations of the layers. As a result of the way in which Transformer models operate, one would expect that a layer is similar to the one following it. This is indeed what is observed globally in Figure 2.2. However, when taking a closer look, some interesting remarks can be made:

- Several rectangular blocks can be spotted. This is confirmed by the curve of the average R_v value which is drawn on top of the heatmaps. Clearly there are breakpoints separating groups of layers that share common characteristics in terms of affinities with other layers.
- One can observe a significant decrease of average similarity on the three last layers with the last layer sometimes having a distinctive behavior.
- ALBERT-large is very different from the other large models in terms layer-wise similarity in the sens that each layer is similar to the previous one and has much

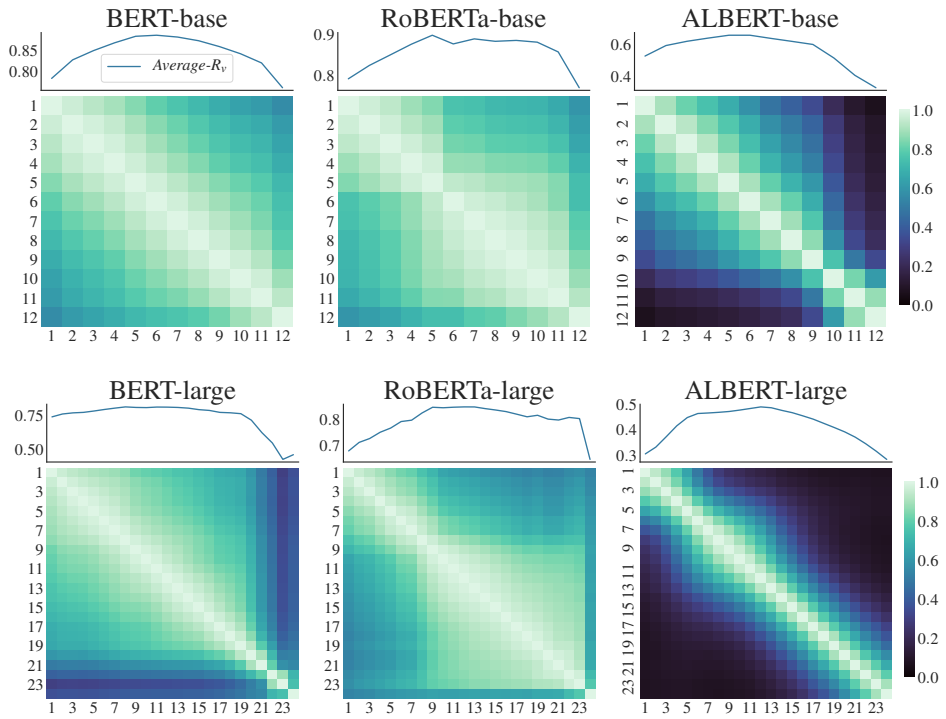

Figure 2.2: R_v-coefficient based layer-wise similarity computed between UFSAC4's data matrices $\mathbf{X}_\ell$.

lower similarity as we move further away in the network. The same happens on the last layers, with not break noticed at the end of the network [1].

Additional insights can be gained from the Spearman correlation coefficients computed on pairs of layers using their ranking vectors $\mathbf{r}_\ell$ (Figure 2.3).

These coefficients allow to refine the observations made on Figure 2.2, we can notice an even bigger difference between the 1st layer and the rest of the network in terms of correlation for BERT-base, moving from $\rho = 0.82$ between layers 1 and 2 to $\rho = 0.95$ between 2 and 3. Overall, Figure 2.3 reveals a certain block structure with groups $\{1\}$, $\{2, 3, 4\}$ and $\{5, 6, 7, 8\}$. Finally, another break can be observed between layers 11 and 12 where $\rho = 0.88$ while it was $\rho = 0.94$ between layers 10 and 11 leading to two new groups $\{9, 10, 11\}$ and $\{12\}$. The same kind of block structure can also be observed when looking at the other models.

[1] This could be explained by the parameter sharing technique used to train the ALBERT model, which consists of duplicating the same parameters for all layers (Lan et al., 2020).

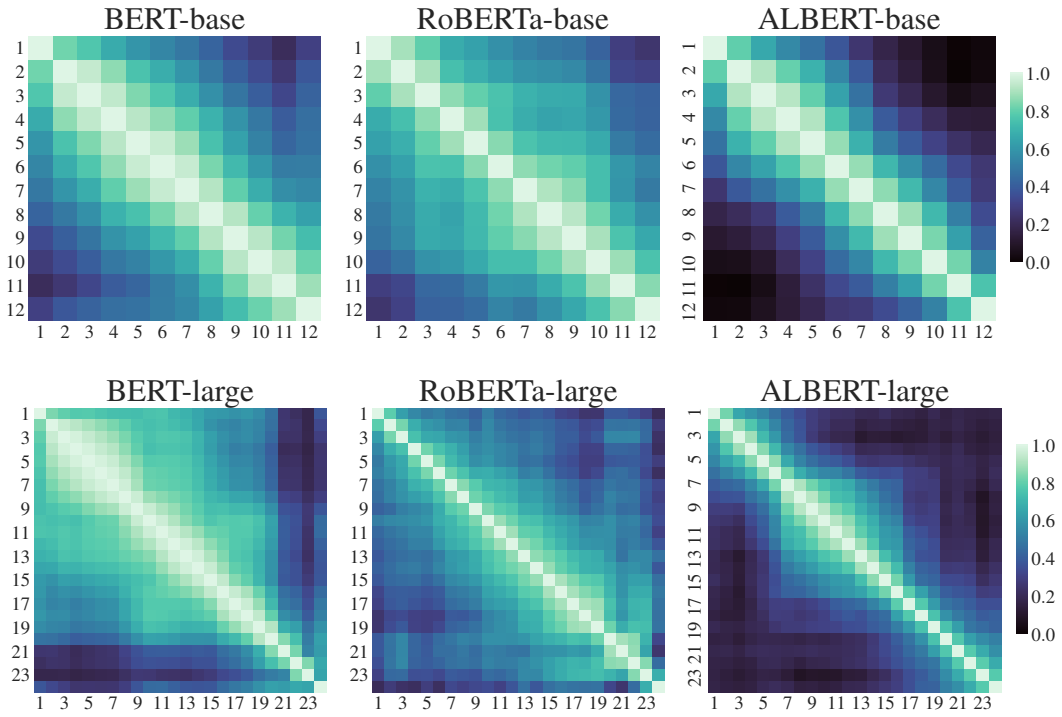

Figure 2.3: Layer-wise agreements using Spearman correlation coefficient: the agreement coefficient between two layers ℓ and ℓ' is the Spearman correlation coefficients ρ calculated between $\mathbf{r}_\ell$ and $\mathbf{r}_{\ell'}$.

2.4.3 Identifying Clusters of Layers

In order to have another look at the possible groupings of layers, we perform an AHC and draw the associated dendrograms (cf. Section 2.3.3). Figure 2.4 shows the results obtained using the Ward and Average linkage criteria, used respectively with the euclidean and cosine distances. If we look at the results for BERT-base that are obtained using the Euclidean distance, we can see that the clusters are close to the groupings suggested by the methods of Section 2.4.2 (compare with the top-left heatmap in Figure 2.3), with the exception of layer 12 which strangely seems to be close to layer 1. This can be explained by the fact that we use the euclidean distance, which tends to be sensitive to the amplitude of data. If we look at BERT-base's box plots in Figure 2.5, it can be seen that the variance of the last layer is very close to that of the 1st layer. To confirm that this wrong assignment was due to an amplitude issue, we experimented with the same AHC algorithm using a cosine distance (known to be insensitive to vector magnitude) with the "average" linkage strategy. With this configuration, the 1st layer is even more separated from the following ones and as expected, the last layer is much less close to the 1st and is assigned to a separate cluster,

which is coherent with the previous observations (Section 2.4.2). This intuition is confirmed when looking at RoBERTa-large, for which we don't have the problem of differing variances across layers (Figure 2.5) and hence presenting almost the same groupings using the two distances. Overall, clustering the layers leads to the following observations:

- As shown in Figure 2.4 for BERT-base, the 4th layer is merged with the $\{2, 3\}$ cluster before the 1st layer, which confirms the break between the first layer and the following ones. In fact, the first layer is, for most models, isolated in its own cluster. This behavior is visible in Figure 2.2 and even more in Figure 2.3 where the 1st layer (1st row) has darker colors than its following neighbors, which indicates lower correlation values compared to the other layers.

- For RoBERTa-large we can also see that the 1st layer joins the 2nd only after layers 3 and 4 (especially in the Cosine version). The last layer is also isolated, joining a cluster only at the 3rd merge of the AHC. The rest of the clusters generally contain successive layers (like $\{5, 6, 7\}$). When cutting at the second merge level we end up with the following partition $\{1\}$, $\{2, 3, 4\}$, $\{5, 6, 7\}$, $\{8, 9, 10\}$, $\{11, 12, 13, 14\}$, $\{15, 16, 17\}$, $\{18, 19, 20, 21\}$, $\{22, 23\}$, $\{24\}$.

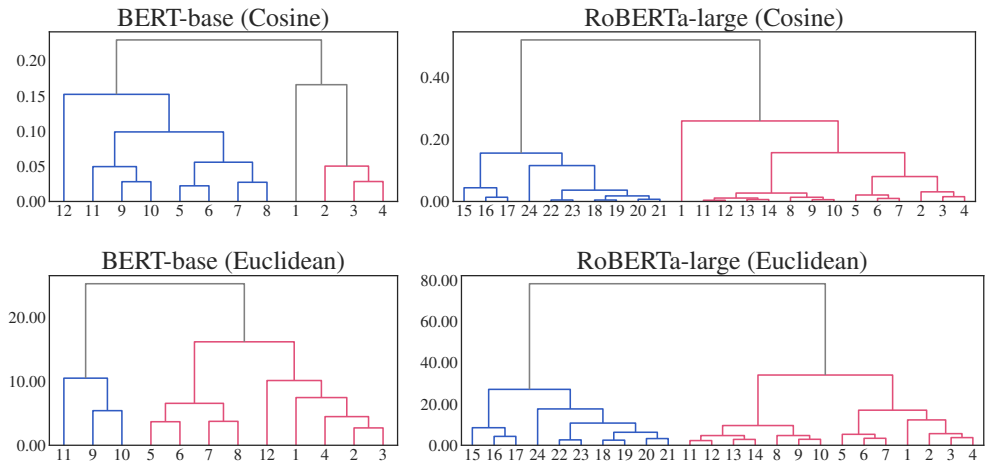

Figure 2.4: Dendrograms obtained with AHC from the set of layers where each layer is represented by $\mathbf{v}_\ell$ a $d-$dimensional vector computed on UFSAC4.

In Figure 2.5, we use the vector representation $\mathbf{v}_\ell$ to draw box plots to analyze the distribution's evolution of layers over the network. We first observe that the three models present different behaviors in terms of variance with from the smallest to the largest: RoBERTa, BERT and ALBERT. Despite that, all distributions are centered around zero with the lower and upper boundaries being quite symmetric. Besides, for BERT (base and large), we can

observe a certain break at the last layers (progressive increase followed by a sudden drop) corresponding to the breaks of similarity observed in Figures 2.2 and 2.3.

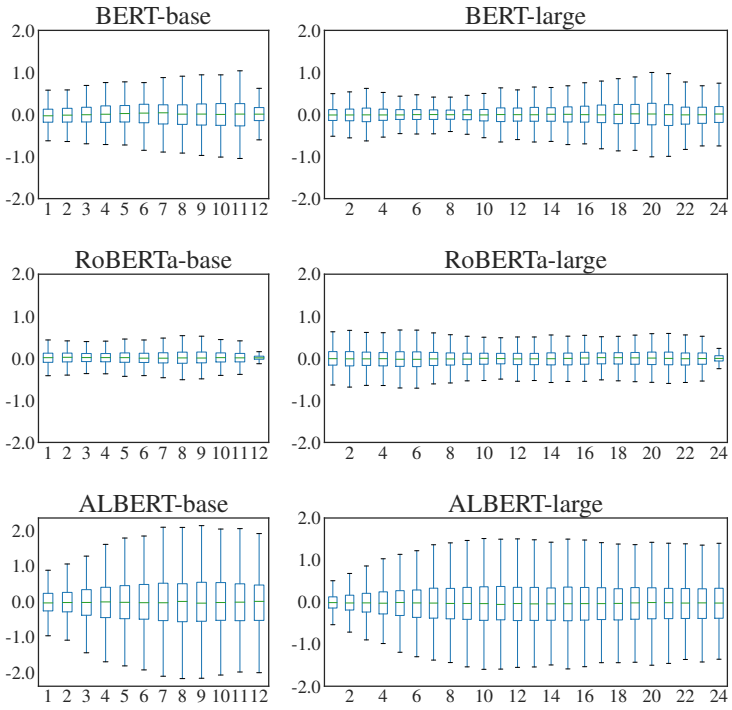

Figure 2.5: Evolution of box plots (without outliers) over layers: each layer is represented by its average vector $\mathbf{v}_\ell$ of the UFSAC4 dataset.

2.4.4 Qualitative Interpretation

Table 2.4 shows the first $m = 30$ words that are the closest to the $\mathbf{v}_\ell$ representations of a selection of layers (due to space limitations) of BERT-base for the UFSAC4 dataset. Confirming the insights provided by the previous rank-based comparisons between layers as well as the clustering experiments, one can note a significant break between layer 1 and its immediate neighbors. Layers 2, 3 and 4 resemble each other more than they resemble layer 1, and share more words such as axons, sclera and scrotum. The correlation scores displayed on top of each pair of layers in Table 2.4 confirm this visual inspection. Between layers 5 and 8 (not shown here), we observed a continuous shift of words in the sense that a limited number of words appeared and disappeared from a layer to another, with the vanishing of Botany words from the 5th layer.

More new words start to appear on the 9th layer like Bermuda, sinus and hepatic with an increasing number of Geography words, until layer 11 inclusive. Layer 12 includes

Table 2.4: Ranking of the words (colored according to their topic) that are closest to $\mathbf{v}_\ell$ representations the BERT-base layers for the UFSAC4 dataset. The first row contains the pairwise Spearman correlation coefficient.

	$\rho = 0.82$	$\rho = 0.95$	$\rho = 0.94$		$\rho = 0.93$	$\rho = 0.94$	$\rho = 0.88$	
1	2	3	4	9	10	11	12
cerebellar	cerebellar	cerebellar	bronchial	adenoids	adenoids	anus	penis
bulbs	kernels	bronchial	cerebellar	atrium	cerebellum	eardrum	eardrum
perennials	maxillae	sclera	molars	hipbones	anus	penis	anus
orchids	bronchial	bronchioles	cranial	anus	Bermuda	armpit	ribs
bronchioles	sclera	clavicles	axons	cerebellum	atrium	cortical	cortical
lymphocyte	cerebellum	cerebellum	arterioles	armpit	eardrum	Bermuda	sphincter
mosses	deserts	cranial	sclera	gyral	armpit	cerebellum	clitoris
rootstocks	clavicles	molars	bronchioles	Bermuda	penis	atrium	pelvic
bronchial	axons	axons	clavicles	sinus	gyral	skull	bulbar
clavicle	bronchioles	arterioles	cerebellum	sphincter	sphincter	Egyptian	calf
leaflets	arterioles	brachial	follicle	leg	Armenia	breastbone	skull
follicle	molars	maxillae	epidermis	clitoris	clitoris	adenoids	armpit
arteriovenous	hindbrain	axon	axon	brachial	pelvic	pelvic	peritoneum
occipital	interface	cervical	brachial	breastbone	hipbones	Bavaria	palmar
maxillae	bulbs	rootstocks	scrotum	incisors	breastbone	clitoris	cerebellum
cheekbone	scrotum	kernels	cervical	skull	calf	Armenia	Carpal
cerebellum	cranial	scrotum	cheekbone	eardrum	skull	gyral	gallbladder
cranial	pods	interface	hindbrain	hepatic	sinus	sphincter	distal
epidermis	sphincter	cerebrum	maxillae	brain	arteriovenous	liver	leg
mucosa	pylorus	deserts	peritoneum	arcuate	Egyptian	calf	wrist
clavicles	rootstocks	epidermis	saliva	cheekbone	leg	peritoneum	hips
pods	epidermis	follicle	incisors	eye	Bavaria	sinus	gut
herbaceous	areolae	sphincter	triceps	muscles	cortical	membrane	bronchial
brachial	follicle	peritoneum	aorta	bones	arcuate	Syria	anal
metatarsal	axon	hindbrain	rootstocks	rump	body	bulbar	scrotum
kernels	glomeruli	saliva	atrium	liver	liver	arcuate	brachial
arterioles	peritoneum	cheekbone	lumbar	cilia	CNS	leg	fibula
pylorus	lymphocyte	triceps	cerebrovascular	Armenia	hepatic	hipbones	Bermuda
metatarsus	arteriovenous	cerebrovascular	gyral	ribs	rump	palmar	liver
molars	occipital	arteriovenous	kernels	dental	cardiovascular	hips	basal

new words that do not appear before like ribs, Carpal and gallbladder and fewer number of Geography words. Overall, the qualitative analysis seems to be in good agreement with what was observed with the previous methods. For example, it seems to attest to the existence of five groups of layers in the standard BERT-base model, namely $\{1\}$, $\{2, 3, 4\}$, $\{5, 6, 7, 8\}$, $\{9, 10, 11\}$ and $\{12\}$.

2.4.5 Quantitative Interpretation Using Dimension Reduction

In this section, relying on PCA, the objective is to go further in the unsupervised analysis of embeddings at different layers. Figure 2.6 presents visual representations provided by PCA applied to the $\mathbf{X}_\ell$ matrices of each layer. First, on the projections of samples, one can observe a significant enhancement of the separability of samples between the layers 1 and 2 whereas it is almost the same between 2 and 3. We noticed another difference between

layers 4 and 5 with a sort of rotation of samples along with a higher increase of variance explained by the two first components. The separability remains more or less stable until layer 11 inclusive and deteriorates in the 12th layer, which also knows a significant increase of explained variance. These differences in separability indicate that the extremities of the network are not only different, but may be much less efficient. Concerning the correlation circles, we notice more differences between layers 1 and 2 than between 2 and 3, this confirms that the 1st layer constitutes a singleton. We also observe a shift across layers with many dimensions that appear in few consecutive layers and then disappear (like 643 appearing in the 2nd layer and disappearing in the 5th layer). Another significant break is observed at the last layer, where dimensions like 223 and 636 disappear while being important for layers 9, 10 and 11. These observations reinforce our previous groupings for BERT-base.

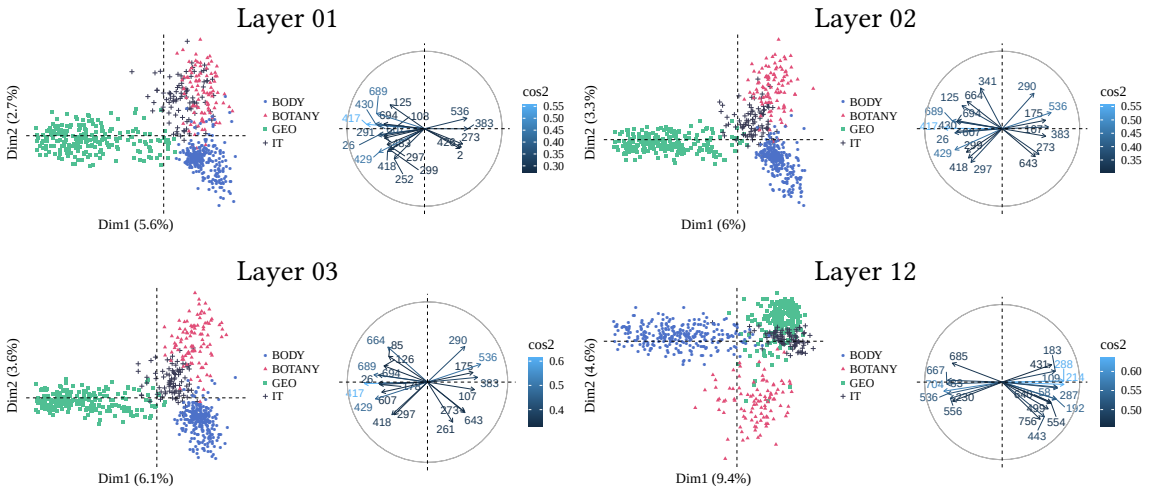

Figure 2.6: PCA on BERT-base's data matrices $\mathbf{X}_\ell$, $\ell = 1, \ldots, b$ - Projections (left): coordinates of words on the two first principal components colored w.r.t. their topic. - Correlation circle (right): only the 20 dimensions that are most correlated with the two first components are displayed.

2.4.6 Results Validation Using a Clustering Performance Metric

In this section, we provide numerical results assessing the layer-wise performance on word clustering using NMI scores (Figure 2.7) on clustering partitions obtained with K-means applied to $\mathbf{X}_\ell$ matrices. In doing so, we aim to validate the layer groups that have been identified in the previous sections. The main question we try to answer is: Do the previously identified groups share characteristics in terms of clustering performance? This study also gives us an idea of the transferability of each layer and each model for the unsupervised

Table 2.5: NMI scores on blocks of layers with UFSAC4 - The first table corresponds to BERT-base and the second to RoBERTa-large. The groups obtained based on word clustering performance fairly closely correspond to the groups that had been spotted using correlation and cluster analyses.

ℓ	01	02	03	04	05	06	07	08	09	10	11	12
NMI	0.64	0.78	0.81	0.88	0.9	0.94	0.9	0.92	0.91	0.91	0.88	0.83

ℓ	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
NMI	0.51	0.57	0.55	0.61	0.81	0.87	0.9	0.7	0.64	0.62	0.63	0.62	0.61	0.62	0.61	0.55	0.53	0.51	0.48	0.41	0.41	0.58	0.57	0.38

task of word clustering. By separating layers into groups based on the NMI scores they achieve, one can find clusters of layers that quite resemble the breakdown suggested by the dendograms in Section 5.3 (compare the values in Table 2.5 with the corresponding dendograms depicted in Figure 2.4). For BERT-base, in the same way as layer 1 is isolated in its own singleton cluster, its NMI score is also the worst. The group formed by 1, 2 and 3 achieves values between 0.78 and 0.88, while the best performers are the layers from 5 to 8. Performance then decreases, with a marked drop at the last layer, again in agreement with the grouping patterns observed in Figures 2.3 and 2.4. The same observations extend to RoBERTa-large where the cluster $\{5, 6, 7\}$ contains the best performing layers. We also clearly see a breaking point of performance between the 1st layer and the following, and another one (more acute) at the last layer. These breaking points are visible in Figures 2.2 and 2.3. In addition, these observations allow us to confirm some findings presented in the supervised study (Liu et al., 2019a) showing that BERT models achieve their best performance on the intermediate layers. We also extend this observation to RoBERTa with fewer well performing layers, situated more earlier in the network.

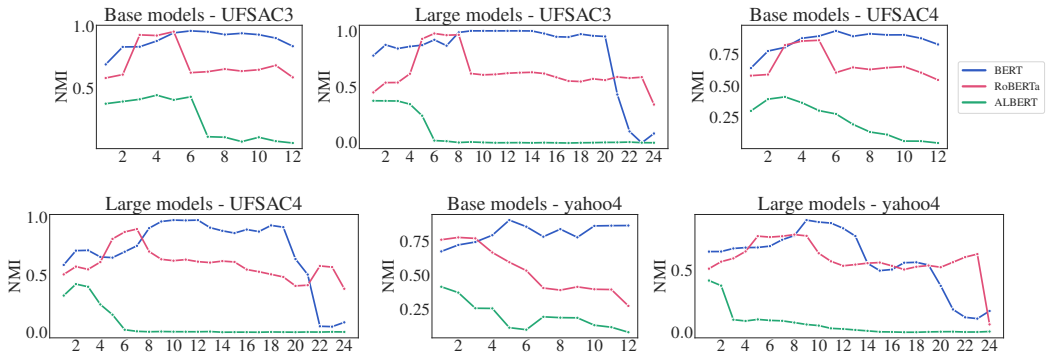

Figure 2.7: NMI scores obtained by the word clustering on the $\mathbf{X}_\ell$ data matrices for each layer ℓ.

More generally, on both base and large versions, BERT outperforms the three other models, followed by RoBERTa and far away by ALBERT. This is surprising considering that ALBERT is supposed to outperform BERT and RoBERTa when fine-tuned on supervised

tasks. We then show that in a no fine-tuning configuration, BERT word embeddings are of higher quality (BERT-large is the only model to achieved the perfect score on UFSAC3). Finally, ALBERT is the only model for which the base version is better than the large one. Moreover, both versions present very poor results on word clustering (especially the large version) and we can notice a better (but still poor) performance with the first layers. One possible explanation is that ALBERT's layers start to be task specific from the beginning of the network, particularly in view of the architecture of ALBERT where all parameters (including attention parameters) are shared across layers.

2.5 Conclusion

Knowing more about contextualized word embeddings and what can really be expected from them is an important topic. In this chapter, we proposed a novel way of analysing Transformer embeddings, based on unsupervised methods, more specifically a correlation and cluster analysis of the layers. Applying these methods to real datasets made it possible to spot precise groups of layers (e.g. 5 groups of layers in BERT-base and 9 in RoBERTa-large) which subsequently proved to fairly closely match the groups obtained when grouping layers based on their clustering performance. This suggests that the proposed method, when applied to a dataset is capable of identifying in advance groups of layers that are likely to best or worst perform on the clustering task. This study also allowed to bring out major differences between Transformer models on the important text clustering task, for example the specificity of ALBERT, which is most likely due to its different network architecture, or the fact that BERT seems to outperform RoBERTa on the clustering task. Future path for research is to further investigate these differences as well as the potential of dimension reduction techniques on contextual word embeddings, an issue that deserves to be the subject of further study, allowing in particular to highlight the potential redundancies present in the Transformer networks.

Chapter 3

Contextual Word Embeddings Clustering through Multiway Analysis

Transformer-based contextual word embedding models are widely used to improve several NLP tasks such as text classification, question answering and named entity recognition. Knowledge about these multi-layered models is growing in the literature, with several studies trying to understand what is learned by each of the layers. However, little is known about how to combine the information provided by these different layers in order to make the most of the deep Transformer models. On the other hand, even less is known about how to best use these modes for unsupervised text mining tasks such as clustering. We address both questions in this chapter, and propose to study several multiway-based methods for simultaneously leveraging the word representations provided by all the layers. We show that some of them are capable to perform word clustering in an effective and interpretable way. We evaluate their performances across a wide variety of Transformer models, datasets, multiblock techniques and tensor-decomposition methods commonly used to tackle three-way data.

3.1 Introduction

Transformer-based contextual word embedding models have revolutionized the NLP state of the art. When fed with a word sequence, a Transformer model produces several different embeddings (one at each layer of the network) for each word in the sequence. Thus, in a word clustering context, for a dataset with n words and a model with b layers and latent dimension d, we can form a 3-way array of shape $n \times b \times d$ (Figure 3.1) where each word is represented by b vectors.

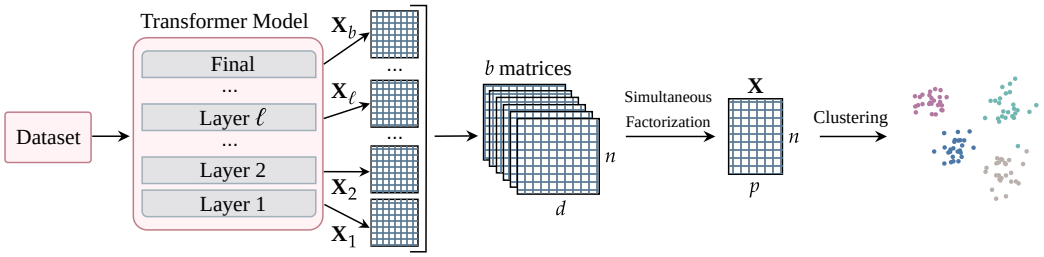

Figure 3.1: Description of the proposed approach that leverages the whole Transformer model by performing multiway clustering using all of the layers' representations.

The information captured at the different layers greatly varies: typically, the early layers may encode local syntactic phenomena while more complex semantic aspects are captured at the higher layers (Tenney et al., 2019a). As a consequence it is no surprise that using as input the embeddings produced at different layers may result in very different performance levels on a given downstream task. Figure 3.2 exemplifies this phenomenon in the case of text clustering.

This problem can be overcome in a supervised context, where it is always possible to train and evaluate the model on all the layers and, using labels, determine the one that maximises a certain metric. In contrast, in the unsupervised setting it is difficult to determine the best layer in advance with no a *priori* information about the data. In the absence of ground truth, since there is no easy way of knowing which layer has given the best result, the solution is to design a technique guaranteeing a performance level better than that provided by the best layer, or at worst better than the mean of all layers's scores. More specifically, as a word described by several embedding vectors can be regarded as an input data observation described by several sets (blocks) of variables, we propose a multiblock approach (Escofier et al., 1983) to make the best use of the information provided by the different layers in a word clustering context. Also, since the data matrices that can be derived from the different layers can be seen as the slices of a 3D-tensor, we also propose an approach based on tensor-decomposition (TD) techniques (Kolda and Bader, 2009). We measure the results of these mutiblock- and tensor-based approaches on a word clustering task, and compare them with the performance obtained either with other ways of aggregating the information contained in all the layers such as *unfolding* (or *matricization*) or with layer-wise, non-multiway clustering methods where each layer is handled separately. To the best of our knowledge, this is the first study that relies on multiway clustering to make the best use of Transformer-based word representations. The main contributions of this study are as follows:

- While the performance delivered by Transformer embeddings is mostly assessed in the context of supervised tasks, we study the behavior of these embeddings in the unsupervised setting, and provide a thorough investigation into how to make the best use of them for word clustering purposes, across a large variety of Transformer models.
- We show that certain multiway methods, simultaneously considering the features provided by all the embedding layers, are capable of delivering a performance level better than that provided by the best layer or at worst better than the mean of all layers's scores.

3.2 Related Work

Several studies focus on explaining how language models like BERT work. Some of them investigate the linguistic features learned by Transformer word embedding models like BERT as in (Tenney et al., 2019a) where it has been showed that early layers encode most local syntactic phenomena while more complex semantics appear at the higher layers. Liu et al. (2019a) evaluated the performance of contextualized word representations on several supervised tasks and compare layers with each other, including ELMo, BERT (base and large) and OpenAI transformer algorithms. They especially observe that Transformers' middle layers present a better transferability. On the other hand, Hao et al. (2019) observed that the early layers of BERT are more invariant across tasks and hence more transferable.

It has also been shown in (van Aken et al., 2019) that, after fine tuning BERT on Question Answering, the model acts in different phases starting from capturing semantic sense of tokens in the first layers to separating the answer token from the others in the last layers. This suggests that the closer we get to the last layer, the more task specific the representations are. This explains the results obtained by Kovaleva et al. (2019) who studied the changes between pre-trained and fine-tuned BERT-base model in terms of attention weights. A significant change in the two last layers in terms of cosine similarity between original and fine-tuned attention weights has been observed on 6 GLUE tasks. The authors deduced that the BERT-base's two last layers learns more task specific features.

The same kind of analysis has not yet been performed for important unsupervised tasks such as clustering. When performing a downstream task using pre-trained Transformers embeddings, an emprirical approach is frequently taken. For example, in (Li et al., 2020b) the authors used pre-trained BERT-base's embeddings to perform document clustering, using the second-to-last layer without real justification. On the other hand, when using popular tools such as "BERT as a service" to build embeddings of text snippets, most practitioners

uses the second last layer as it is the default layer used by the service (the last one deemed to be biased towards the Masked Language Model training function).

We show that this choice of layer is not optimal for a clustering task, where the best performance is most often achieved by using the middle layers, especially for large Transformer models (see Figure 3.2). Moreover, it will even be shown (see Figure 3.2) that, if post-processed using a standard, the last layer, which is in indeed quite useless in its raw form, may become an extremely valuable input for a word clustering task. More importantly, since, unlike syntactic information which is generally concentrated in a few layers, semantic features are spread out across the entire network (Tenney et al., 2019a), and since both syntactic and semantic features can be helpful in word and text clustering, the conclusion is that one should try to benefit from all the representations provided by Transformer models to perform unsupervised learning.

Some papers have considered the use of multiway techniques in text-mining applications like Kolda et al. (2005) who proposed to use a CP-like method on sparse three-way web data where the three modes are webpages $\times$ webpages $\times$ anchor text, allowing to identify topics while grouping webpages. In text classification, Liu et al. (2005) proposed to represent each document by a character level tensor $\boldsymbol{\mathcal{T}}$ of size $27 \times 27 \times 27$ considering 27 as the size of the vocabulary (including the 26 letters and '_' for the rest of characters) where each value $\boldsymbol{\mathcal{T}}_{ijk}$ corresponds to a tri-gram formed by the ith, jth and kth characters of the vocabulary. Overall, these studies, focusing on tensor-decomposition techniques, first are far from having explored the broad spectrum of possibilities offered by the mutltiway techniques, and second they all predate the appearance of Transformer models.

3.3 Word Clustering Using Transformer Embeddings

For a given dataset of n words, and a model with b embedding layers, we obtain b different raw representations of size d, one from each layer. The dataset can then be represented by b matrices $\mathbf{X}_1, \ldots, \mathbf{X}_b$ of size $n \times d$ as shown in Figure 2.1.

One can then choose to use each of these matrices individually, the problem being that in an unsupervised context it is not possible to determine in advance which matrix is likely to help achieve the best performance. One alternative is to try to benefit from the information provided by all the layers, simultaneously. We therefore investigate different clustering methods, taking as input, for a given dataset:

1. Each raw matrix $\mathbf{X}_\ell$ in isolation from the other (obtaining one clustering partition per layer).

2. A dimension-reduced version of each raw matrix $\mathbf{X}_\ell$, using PCA (obtaining one clustering partition per layer as in the previous case).

3. A matricized version of the $\mathbf{X}_\ell, \ell = 1, \ldots, b$ matrices, where these matrices are concatenated horizontally into a matrix of size $n \times (b \times d)$ (obtaining a single clustering partition).

4. A PCA-reduced version of the matricization of the $\mathbf{X}_\ell, \ell = 1, \ldots, b$ matrices (obtaining a single clustering partition).

5. The p common components obtained with all of the $\mathbf{X}_\ell, \ell = 1, \ldots, b$ matrices as blocks given as input to the multiblock methods described in Section 3.3.3 (obtaining a single clustering partition).

6. The p mode-1 factors that compose the matrix $\mathbf{A}$ obtained with all of the $\mathbf{X}_\ell, \ell = 1, \ldots, b$ matrices as slices of a three-way tensor given as input to tensor-decomposition methods, as described in Section 3.3.4 (obtaining a single clustering partition).

Using each layer's output separately result in b different clustering partitions. As already said, considering the unsupervised setting of our task and the absence of true labels, it is impossible to determine in advance the best performing layer. In addition, it is worth noting, as demonstrated by the experiments described later in the chapter, that it is impossible to determine a unique layer for all datasets, since the best layer highly depends on the dataset used, and there is no satisfying universal choice. Therefore, we propose an alternative that benefits from all the network's layers, which we show to be very efficient while freeing us from the impossible task of choosing a unique layer to perform clustering.

3.3.1 Layer-wise Clustering

In layer-wise clustering, we use the K-means algorithm and give it as input a matrix $\mathbf{X}_\ell$ of n rows, corresponding to the output of the ℓth layer of the model, and eventually post-processed using PCA. The PCA-reduced representations are formed by the p first principal components of $\mathbf{X}_\ell$. Since we cannot tune the p parameter in our unsupervised setting, we set it to the same value ($p = 15$) for all experiments.

For a model with b layers, one can think of running the clustering algorithm on each of the b $\mathbf{X}_\ell$ matrices, and then pick the one that yields the "best" result. However, in the unsupervised setting where no labels are available, there is no easy way of knowing which layer $\mathbf{X}_\ell$ has given the best result. The solution is to use a technique that frees us from the impossible task of choosing the best layer while guaranteeing a performance level

better than that provided by the best layer or at worst better than the mean of all layers's scores. As will be shown in the experiments section, an Ensemble approach can meet these requirements.

In layer-wise clustering, we use the K-means algorithm with as input a matrix $\mathbf{X}_\ell$ of n rows provided by the ℓth layer of the model. We also perform clustering on the PCA-reduced representations, that are formed by the p first principal components of $\mathbf{X}_\ell$. Layer-wise clustering is referred to as LW when using the raw embeddings and LW-PCA when using the post-processed versions with PCA.

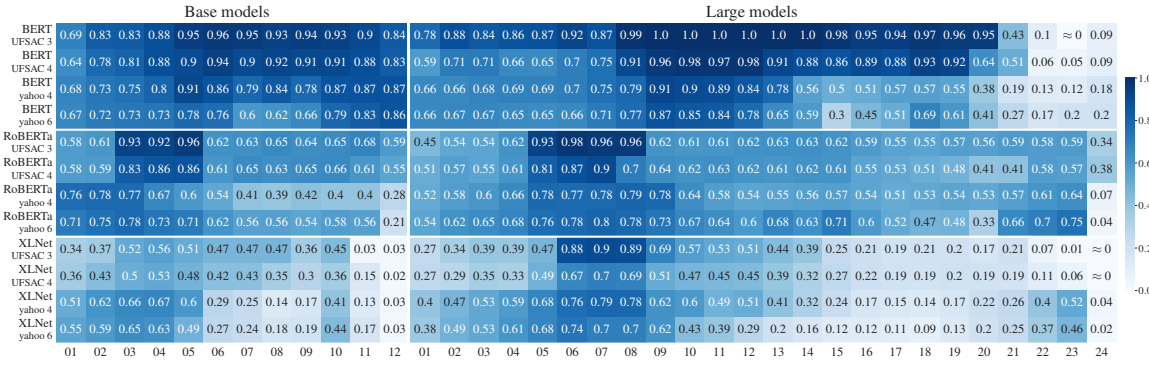

Figure 3.2: Layer-wise clustering performance results: NMI scores. The obtained performance highly depends on the layer which is used as input to the clustering task.

3.3.2 Unfolding Layers' Representations

To try to unleash the potential of all the layers, a simple approach consists in concatenating the matrices $\mathbf{X}_\ell, \ell = 1, \ldots, b$ into a unique data matrix $\mathbf{X}$ of size $n \times (b \times d)$. For example, given a base model with 12 layers and 768 dimensions which provides 12 matrices $\mathbf{X}_b$ of size $n \times 768$, we obtain after the unfolding a matrix of size $n \times 9\,216$. We call UN the use of $\mathbf{X}$ directly by a clustering algorithm. We also perform clustering after reducing $\mathbf{X}$ using PCA and call this approach UN-PCA.

3.3.3 Multiblock Analysis

With a view to taking advantage of each layer of a given Transformer language model, we propose to harness multiblock –or multitable– methods: Consensus PCA (CPCA) (Wold et al., 1987), Generalized Canonical Correlation Analysis (GCCA) (Carroll, 1968), Multiple Co-Inertia Analysis (MCIA) (Chessel and Hanafi, 1996), Multiple Factor Analysis (MFA)

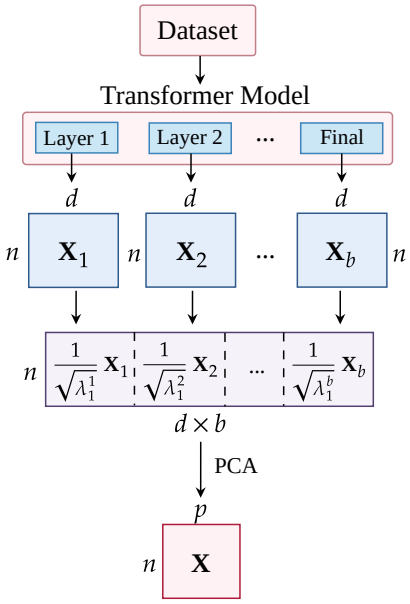

Figure 3.3: Construction of the multiblock reduced data matrix using MFA, λ_1^ℓ being the first eigenvalue of $\mathbf{X}_\ell$.

(Abdi et al., 2013; Escofier et al., 1983), STATIS (Escoufier, 1980), Common Components and Specific Weights Analysis (CCSWA) (Qannari et al., 2000). These methods are designed to deal with simultaneous dimensionality reduction in b blocks (with different features describing the same observations) and are particularly popular for analyzing multi-omics data (Argelaguet et al., 2018; Li et al., 2012) as well as sensory profiling data (Llobell et al., 2020; Wilderjans and Cariou, 2016). Thereby we argue that they can be very useful to tackle word clustering from three-way data.

In our case, we consider each layer ℓ as a block and each corresponding data matrix is represented by $\mathbf{X}_\ell$ of size $n \times d$ and $\mathbf{S}_\ell = \mathbf{X}_\ell \mathbf{X}_\ell^T$, where n is the number of samples in the dataset and d the number of features of the word representations (d is the same for all of the layers). The objective is to represent the b blocks by a unique matrix $\mathbf{W}$ of size $n \times p$ formed by p component vectors $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_p$ each one of size $n \times 1$ that optimally resumes the overall information present in the blocks.

The MFA method can be seen as an extension of PCA adapted to multiblock data. It consists in applying a standard PCA on a data matrix $\mathbf{X}$ whose features are composed of all the variables (dimensions) weighted according to the block (layer) they belong to in order to balance the influence of each block of variables. The balance is achieved by normalizing each data block $\mathbf{X}_\ell$ using the first eigenvalue λ_1^ℓ. $\mathbf{X}$ is then obtained by concatenating the b resulting matrices. Another formulation of the problem is finding the vector $\mathbf{q}_1$ that is the

most linked to all the weighted variables. More formally, MFA maximizes:

$$\sum_{\ell=1}^{b} \mathcal{L}_\ell(\mathbf{X}_\ell, \mathbf{q}_1) = \sum_{\ell=1}^{b} \frac{1}{\lambda_1^\ell} \sum_{j=1}^{d} cov^2(\mathbf{x}_{\ell_{.j}}, \mathbf{q}_1) \tag{3.1}$$

where $\mathbf{x}_{l_{.j}}$ corresponds to the jth variable of the data block $\mathbf{X}_\ell$ and $\mathbf{q}_1$ to the first component of MFA. The next component maximizes the same criterion while being orthogonal to $\mathbf{q}_1$.

The STATIS method is similar to MFA, and differs in the weighting step. With STATIS, instead of weighting the data blocks according to their corresponding eigenvalues, each data block is weighted using the first eigenvector $\mathbf{v}_1$ of the inner product matrix C of size $b \times b$ where each element $c_{\ell,\ell'}$ is computed as follows:

$$c_{\ell,\ell'} = trace(\mathbf{S}_\ell \times \mathbf{S}_{\ell'}) = \sum_{i=1}^{n} \sum_{k=1}^{n} s_{i,k,\ell} s_{i,k,\ell'}. \tag{3.2}$$

The weighting vector $\boldsymbol{\alpha}$ is computed by $\boldsymbol{\alpha} = \mathbf{v}_1 \times (\mathbf{v}_1^T \mathbb{1})^{-1}$ so that the elements of $\boldsymbol{\alpha}$ sum to one ($\mathbb{1}$ being the unit vector). In (Hanafi et al., 2011), the authors discussed the difference between CPCA, MCIA and GCCA and show their similarity, which lies in the optimized criterion that aims to reveal covariant patterns in and between the different blocks by finding scores vectors $\mathbf{q}_1^\ell$ for each block ℓ which are as much linked as possible to a global score vector $\mathbf{q}_1$: $\sum_{\ell=1}^{b} cov^2(\mathbf{q}_1^\ell, \mathbf{q}_1)$ where $\mathbf{q}_1^\ell$ and $\mathbf{q}_1$ are $n \times 1$ vectors. This criterion is maximized by the three multiblock methods, with different constraints. The same function is maximized to find the higher order components, using a deflated version of the current data blocks at each iteration. The deflating function used with CPCA is as follows:

$$\mathbf{X}_\ell^{(t+1)} = \mathbf{X}_\ell^{(t)} - \frac{\mathbf{q}_t \mathbf{q}_t'}{\mathbf{q}_t' \mathbf{q}_t} \mathbf{X}_\ell. \tag{3.3}$$

The CCSWA (Qannari et al., 2000) method (sometimes called ComDim) also computes the components iteratively. The first component $\mathbf{q}_1$ and their weights $\gamma_1^1, \ldots, \gamma_1^b$ are computed as to minimize the expression: $\sum_{\ell=1}^{b} \left\| \mathbf{S}_\ell - \gamma_1^\ell \mathbf{q}_1 \mathbf{q}_1^T \right\|^2$ where γ_1^ℓ represents the salience value of the ℓth block for the first component (common axis) represented by the vector $\mathbf{q}_1$. The CCSWA algorithm aims to find the parameters $\gamma_t^1, \ldots, \gamma_t^b$ and $\mathbf{q}_t$ for $t = 1, 2, \ldots, p$ that maximizes an overall loss function at each iteration t:

$$\sum_{\ell=1}^{b} ||\mathbf{S}_\ell - \sum_{k=1}^{t} \gamma_1^\ell \mathbf{q}_k \mathbf{q}_k^T||^2. \tag{3.4}$$

This proportion belongs to $[0, 1]$ and the closer to 1 it is, the better is $\mathbf{X}_{\ell'}$ as a substitute for $\mathbf{X}_\ell$ (and *vise-versa*) to characterize the n samples of the dataset.

Whatever the method used, the p first common components $\mathbf{q}_1, \ldots, \mathbf{q}_p$ are used as input to a clustering algorithm as shown in Algorithm 2, thus leveraging all of the word representations provided by the multi-layered Transformer models. The common components can be obtained using the R package *FactoMineR* (Lê et al., 2008) for MFA and *mogsa* (Meng, 2019) for CPCA, GCCA, MCIA, and STATIS. For CCSWA, we used our Python implementation.

Algorithm 2: Multiblock Clustering Procedure

Input: a dataset $\mathcal{D}$ of n words; a Transformer model $\mathcal{M}$ of b layers; a clustering algorithm $\mathcal{C}$; a multiway decomposition function $\mathcal{F}$, number of components p
Output: a clustering partition $\mathbf{p}$
Step 1. Build data matrices, for each $\ell = 1 \ldots b$:
$\mathbf{X}_\ell \leftarrow \mathcal{M}(\mathcal{D}, \ell)$ as in Figure 2.1;
Step 2. Perform multiblock factorial decomposition:
$\mathbf{q}_1, \ldots, \mathbf{q}_p \leftarrow \mathcal{F}([\mathbf{X}_1, \ldots, \mathbf{X}_b], p)$;
$\mathbf{W} \leftarrow$ horizontal stacking of the p components $\mathbf{q}_1, \ldots, \mathbf{q}_p$;
Step 3. Perform clustering:
$\mathbf{p} \leftarrow \mathcal{C}(\mathbf{W})$;
return $\mathbf{p}$

3.3.4 Tensor Decomposition

In this family of methods, data matrices $\mathbf{X}_\ell, \ell = 1, \ldots, b$ can be viewed as a three-way or three-modal tensor $\mathbf{X}$ of size $n \times d \times b$ where each matrix $\mathbf{X}_\ell$ (obtained with the ℓth layer of the model) is considered as a slice of the tensor (cf. Figure 3.4). Two of the most popular TD techniques are CANDECAMP/PARAFAC (CP) (Carroll and Chang, 1970; Harshman et al., 1970) and Tucker decomposition (Tucker, 1966). For the sake of simplicity, we describe these two techniques for three-mode tensors, but they can be generalized for m-mode tensors ($m > 3$). A CP decomposition of rank p aims to find three factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ of size $n \times p$, $d \times p$ and $b \times p$ respectively, and $\boldsymbol{\lambda}$ that minimizes the approximation error of $\mathbf{X}$ by finding:

$$\min_{\mathbf{X}^*} \|\mathbf{X} - \mathbf{X}^*\| \text{ where } \mathbf{X}^* = \sum_{j=1}^{p} \boldsymbol{\lambda}_j \mathbf{a}_j \circ \mathbf{b}_j \circ \mathbf{c}_j \tag{3.5}$$

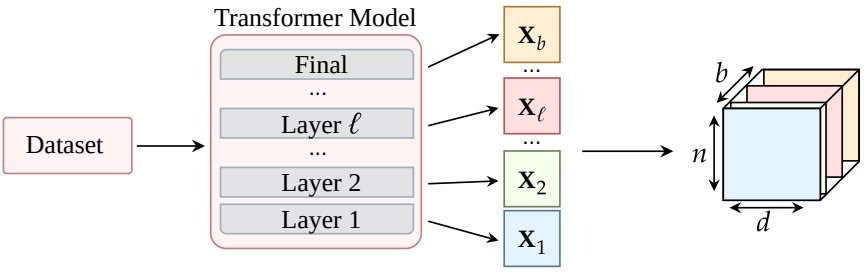

Figure 3.4: Construction of a tensor to be used for multiway clustering with TD techniques.

where $\mathbf{a}_j$, $\mathbf{b}_j$ and $\mathbf{c}_j$ being the jth column of $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ respectively. The "$\circ$" stands for the outer product between two vectors. The outer product of the three vectors $\mathbf{a}_j$, $\mathbf{b}_j$ and $\mathbf{c}_j$ results in a three-way tensor with the sames dimensions as $\mathbf{X}$. One of the most popular procedures to optimize this criterion is the Alternating Least Squares (ALS) (Carroll and Chang, 1970; Harshman et al., 1970). The Tucker decomposition also computes three factor matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$, this time of size $n \times p$, $d \times q$ and $b \times t$ respectively. Unlike in CP, the ranks of the three modes (p, q and t) are not necessarily equal. In addition to the factor matrices, a core tensor $\mathbf{G}$ of size $p \times q \times t$ is computed so as to optimize:

$$\min_{\mathbf{X}^*} \|\mathbf{X} - \mathbf{X}^*\| \text{ where } \mathbf{X}^* = \sum_{i=1}^{p}\sum_{j=1}^{q}\sum_{k=1}^{t} g_{ijk}\, \mathbf{a}_i \circ \mathbf{b}_j \circ \mathbf{c}_k \tag{3.6}$$

g_{ijk} being the intersection if the ith row (mode 1), the jth column (mode 2) and the kth tube (mode 3) of the tensor $\mathbf{G}$. To solve this optimization problem, a popular approach named Higher-Order Orthogonal Iteration (HOOI) (De Lathauwer et al., 2000) and similar to ALS consists in fixing two factor matrices to compute the third one by using two-way singular vector decomposition (further details can be found in (De Lathauwer et al., 2000)).

In our experiments, we use for both CP and Tucker the matrix $\mathbf{A}$ as input to the clustering algorithm. The p columns of $\mathbf{A}$ can be seen as the principal components in the first mode of $\mathbf{X}$. We use two Python implementations of ALS and HOOI to perform CP and Tucker decomposition respectively. The first implementation is provided by the *TensorD* (Hao et al., 2018) package, based on the *TensorFlow* framework, allowing GPU acceleration. The second is available in the *TensorLy* (Kossaifi et al., 2019) package which flexibly leverages several CPU and GPU backends.

3.4 Experimental Study

3.4.1 Datasets and Models Used

The datasets used are described in Table 3.1 by the words, their contexts and the corresponding topic. The first dataset, denoted as UFSAC3 is extracted from the UFSAC dataset (Vial et al., 2018) which consolidates multiple popular WSD datasets (such as SemEval and SensEval) into a uniform format. The examples are manually divided into three topics: Body, Botany and Geography. The second dataset, denoted as UFSAC4, is a slightly more difficult dataset; it includes a fourth class (words related to information technology) and is augmented with some polysemous examples (such as "lobes" which appears in Body and Botany with different contexts). The two last datasets yahoo4 and yahoo6, are extracted from the Yahoo! Answers dataset (Zhang et al., 2015) by manually selecting sets of words for each category and some corresponding contexts.

Table 3.1: Datasets description: k denotes the number of clusters.

datasets	n	k	clusters' sizes
UFSAC3	583	3	body: 266, geo: 227, botany: 90
UFSAC4	691	4	body: 275, geo: 227, botany: 99, IT:90
yahoo4	528	4	finance: 150, music: 82, science-maths.: 152, computers-internet: 144
yahoo6	852	6	health: 201, finance: 150, computers-internet: 144, music: 82, science-maths.: 152, sports: 123

From each set of documents, we compute multiple contextual representations from 6 pre-trained models which are the base (12 layers) and large (24 layers) versions of BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b) and XLNet (Yang et al., 2019b) with a vocabulary size of 28,996 for BERT, 50,265 for RoBERTa and 32,000 for XLNet.

3.4.2 Clustering Evaluation

In all the experiments, we used K-means (Lloyd, 1982) with a known number of clusters (any other clustering algorithm can be used), which are run with 10 different initializations on different matrix representations of words occurrences, surrounded by a given context.

To validate the results produced by the clustering algorithm we relied on standard external measures devoted to assessing cluster quality, namely Normalized Mutual Information (NMI) (Strehl and Ghosh, 2002), the Adjusted Rand Index (ARI) (Hubert and Arabie, 1985; Steinley, 2004) and the clustering accuracy. When performing dimension reduction, a number of components p has to be fixed. Since we cannot tune the p parameter in our unsupervised setting, we set it to the same value ($p = 15$) for all experiments. This corresponds either to the number of principal components of PCA (cf. Sections 3.3.1 and 3.3.2), the number of common components present in the $\mathbf{W}$ matrix computed by the multiblock techniques (cf. Section 3.3.3) or the rank of the matrix $\mathbf{A}$ computed by the TD methods (cf. Section 3.3.4). Table 3.2 contains the NMI scores obtained by each method over the 6 Transformer models and the 4 datasets. For comparison purposes, we also included the results obtained when using fastText word embeddings or the representations obtained by the Transformers' layers individually (as described in the next section). FastText (Bojanowski et al., 2017) was used as a representative of the Word2Vec family of embeddings since it provides for the Out-of-Vocabulary issue which otherwise would distort comparisons. The suffixes TensD and TensL in Table 3.2 stand for the packages *TensorLy* and *TensorD* respectively.

3.4.3 Layer-wise Clustering Experiments

To have an idea on the performance obtained by each layer, we perform layer-wise clustering and present the results (NMI) in Figure 3.2.We also perform clustering on the PCA-reduced representations and present the results (NMI) in Figure 3.5 only for two datasets, due to the lack of space. In Table 3.2, we call LW and LW-PCA the average of the scores obtained by all of the b clustering experiments run on each matrix $\mathbf{X}_\ell$ separately before and after applying PCA respectively. We also present the highest of the layers' scores in order to compare it to the multiway approaches scores, even if these layers are unfortunately impossible to determine in the absence of labels. We call this score BE-LW when using the raw representations and BE-PCA when using the reduced version.

3.4.4 Multiway Clustering Experiments

Since the obtained results may greatly vary with each layer, as clearly visible in Figure 3.5, and since determining which one is the best is impossible in the absence of labels when dealing with real datasets, a sensible solution to overcome this incertitude is to use multiway techniques as described in Sections 3.3.2, 3.3.3 and 3.3.4 using the $\mathbf{X}_\ell$ data matrices $\ell = 1, \ldots, b$ of each model. Table 3.3 provides an overview of the overall performance obtained by the multiway techniques with the mean rank of each method. The mean rank corresponds

Table 3.2: NMI scores of multiway clustering on Transformer and fastText embeddings.

Datasets	Method	BERT-base	BERT-large	RoBERTa-base	RoBERTa-large	XLNet-base	XLNet-large	fastText
UFSAC3	LW	0.89±0.07	0.81±0.3	0.71±0.14	0.64±0.16	0.38±0.17	0.38±0.26	0.92
	LW-PCA	0.91±0.07	0.8±0.31	0.78±0.16	0.66±0.16	0.48±0.12	0.4±0.24	0.92
	BE-LW	0.96	1	0.96	0.98	0.56	0.9	-
	BE-PCA	0.98	1	0.96	0.98	0.8	0.91	-
	UN	0.96	**1.0**	0.94	0.66	0.04	0.0	-
	UN-PCA	**0.98**	**1.0**	**0.97**	0.65	0.54	**0.55**	-
	MFA	**0.98**	**1.0**	**0.97**	0.77	0.55	0.54	-
	STATIS	0.96	0.98	**0.97**	0.77	0.55	0.5	-
	CPCA	0.76	0.71	0.68	0.53	0.56	0.48	-
	GCCA	0.8	0.76	0.74	0.65	0.61	0.52	-
	MCIA	0.7	0.76	0.72	0.69	0.5	0.49	-
	CCSWA	0.58	0.58	0.77	0.68	**0.63**	0.47	-
	CP-TensD	0.66	0.28	0.01	0.64	0.44	0.26	-
	CP-TensL	0.92	0.66	0.57	0.56	0.36	0.02	-
	Tuck-TensD	0.68	0.75	0.78	**0.83**	0.61	0.48	-
	Tuck-TensL	0.73	0.81	0.89	0.69	0.47	0.49	-
UFSAC4	LW	0.86±0.08	0.72±0.28	0.67±0.11	0.6±0.13	0.36±0.14	0.33±0.19	0.77
	LW-PCA	0.88±0.06	0.75±0.25	0.74±0.1	0.63±0.11	0.44±0.06	0.39±0.18	0.84
	BE-LW	0.94	0.98	0.86	0.9	0.53	0.7	-
	BE-PCA	0.95	0.96	0.91	0.89	0.55	0.69	-
	UN	0.96	0.97	0.67	0.65	0.16	0.01	-
	UN-PCA	**0.97**	**0.99**	0.76	0.65	0.47	0.5	-
	MFA	**0.97**	**0.99**	0.73	0.67	0.47	0.49	-
	STATIS	0.96	0.95	**0.9**	0.68	0.47	0.48	-
	CPCA	0.75	0.84	0.75	0.59	0.59	0.53	-
	GCCA	0.76	0.75	0.82	0.7	0.58	0.46	-
	MCIA	0.81	0.76	0.79	0.67	0.56	0.49	-
	CCSWA	0.73	0.82	0.76	**0.76**	**0.61**	0.28	-
	CP-TensD	0.43	0.59	0.06	0.46	0.04	0.44	-
	CP-TensL	0.61	0.55	0.48	0.61	0.41	0.39	-
	Tuck-TensD	0.75	0.85	0.84	0.62	0.51	**0.58**	-
	Tuck-TensL	0.81	0.86	0.79	0.65	0.49	0.55	-
yahoo4	LW	0.81±0.07	0.59±0.24	0.54±0.17	0.59±0.14	0.37±0.22	0.43±0.22	0.42
	LW-PCA	0.83±0.07	0.58±0.25	0.84±0.04	0.66±0.15	0.54±0.24	0.53±0.23	0.51
	BE-LW	0.91	0.91	0.78	0.79	0.67	0.79	-
	BE-PCA	0.92	0.94	0.9	0.89	0.89	0.86	-
	UN	0.9	0.82	0.63	0.71	0.03	0.04	-
	UN-PCA	**0.93**	**0.91**	**0.91**	0.82	0.74	0.73	-
	MFA	0.92	**0.91**	**0.91**	**0.83**	0.73	0.74	-
	STATIS	0.91	0.89	0.9	0.82	0.71	**0.75**	-
	CPCA	**0.93**	0.74	0.87	0.82	0.76	0.66	-
	GCCA	0.88	0.71	0.85	0.79	0.63	0.73	-
	MCIA	0.85	0.76	0.86	0.8	0.67	0.7	-
	CCSWA	0.7	0.73	0.81	0.8	**0.89**	0.62	-
	CP-TensD	0.04	0.08	0.17	0.07	0.41	0.38	-
	CP-TensL	0.44	0.55	0.41	0.63	0.18	0.55	-
	Tuck-TensD	0.81	0.66	0.81	0.79	0.72	**0.75**	-
	Tuck-TensL	0.83	0.73	0.85	0.8	0.8	0.7	-
yahoo6	LW	0.73±0.08	0.58±0.21	0.61±0.15	0.62±0.16	0.37±0.2	0.37±0.22	0.42
	LW-PCA	0.73±0.06	0.57±0.22	0.79±0.08	0.69±0.14	0.51±0.19	0.45±0.24	0.42
	BE-LW	0.86	0.87	0.78	0.8	0.65	0.74	-
	BE-PCA	0.87	0.87	0.93	0.81	0.85	0.78	-
	UN	0.92	0.8	0.72	0.79	0.11	0.04	-
	UN-PCA	0.94	0.81	0.81	0.81	0.65	0.66	-
	MFA	0.93	0.83	0.81	0.81	0.65	0.67	-
	STATIS	0.74	0.75	0.8	0.81	0.66	0.69	-
	CPCA	0.94	**0.9**	0.88	**0.84**	0.69	0.7	-
	GCCA	**0.96**	0.85	0.88	0.83	0.65	0.69	-
	MCIA	0.93	0.86	**0.9**	0.83	0.68	0.69	-
	CCSWA	0.82	0.7	0.65	0.76	**0.85**	**0.77**	-
	CP-TensD	0.53	0.18	0.09	0.11	0.31	0.21	-
	CP-TensL	0.6	0.74	0.57	0.46	0.14	0.3	-
	Tuck-TensD	0.87	0.71	0.78	0.81	0.82	0.65	-
	Tuck-TensL	0.85	0.66	0.79	0.81	0.8	0.64	-

Table 3.3: Mean ranks of multiblock and tensor-based methods (the lower the better). The suffixes Ly and D stand for the packages TensorLy and TensorD respectively.

	UN	UN-PCA	MFA	STATIS	CPCA	GCCA	MCIA	CCSWA	CP-D	CP-Ly	Tuck-D	Tuck-Ly
NMI	11.02	5.6	**4.85**	5.92	6.71	6.88	6.79	9.04	15.12	13.71	8.25	8.12
ARI	10.73	5.06	**4.65**	5.79	7.75	7.42	7.58	8.46	14.88	13.5	8.17	8.12
ACC	11.27	5.73	**5.27**	6.21	7.98	7.21	7.04	8.02	14.5	13.5	8.08	7.88

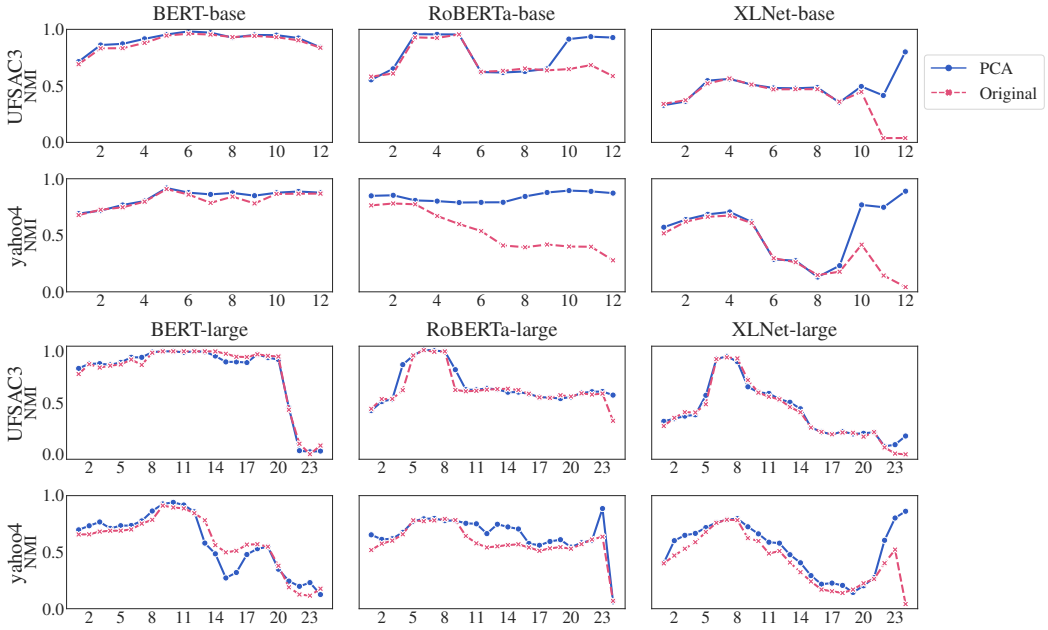

Figure 3.5: Compared clustering performance (NMI) across layers of the original representations of base models (using all of the 768 dimensions of the $\mathbf{X}_\ell$ matrices) vs. reduced representations (using 15 principal components of $\mathbf{X}_\ell$), with $\ell = 1, \ldots, b$ ($b \in \{12, 24\}$).

to the average of all the ranks assigned to a given method based on its performance compared to the other methods, over all the model-dataset combinations. Table 3.2 gives further details of the performance of each method over all the models and datasets.

3.5 Discussion and Interpretation

In this section, we discuss the benefit of applying multiway methods in the unsupervised setting due to a diversity of the information provided by each layer. Thereby, from the representations of each layer used separately we first evaluate the performance in terms of word clustering. We aim to show that the results might greatly vary according to the dataset.

To tackle this issue, for each dataset we propose to use all layers. We perform a dimensionality reduction in different ways and from a simplified representation, a clustering algorithm is applied and quality of word clustering is measured. Further, using user-friendly visualization tools, we show how the classes are arranged in a low-dimensional space, how detecting the most influential words and measuring the role of each layer.

3.5.1 Layer-wise Clustering Results

Figures 3.2 and 3.5 show that BERT presents better performance than RoBERTa and XLNet, even if the latter are more recent and outperforms BERT in the supervised tasks. We can see in the same figures that BERT models achieve their best performance on the intermediate layers (which is in line with the findings issued in (Liu et al., 2019a)). Especially, BERT-large reaches a perfect NMI score on UFSAC3 from the 9th to the 14th layer, with a sudden drop of performance on the 5 last layers. Overall, we can see this decrease for all the models. Figure 3.5 also shows that reducing the number of dimensions to only 15 (which constitutes less than 2% of features for the base models and 1.5% for the large ones) leads to very competitive performance scores, achieving even better results than raw representations, which indicates a high redundancy in the pre-trained embeddings. Interestingly, in the case of RoBERTa and XLNet, the PCA reduction significantly improves the performance of the last layers, which as said previously perform very poorly in their raw form. For example, on yahoo4, the raw vectors provided by the last layer of XLNet-base yield an NMI of 0.03 while they achieve 0.89 with PCA, moving the last layer from the worst to the best layer.

3.5.2 Multiway Clustering Results

One first observation from Table 3.2 concerning the multiway clustering results is that multiblock methods seem more effective compared to TD techniques. This is confirmed by Table 3.3 where MFA has the best mean rank, followed by UN-PCA and STATIS. Furthermore, Tucker seems more suitable for our data than CP. Besides, Table 3.2 shows that in the case of BERT (base and large), the multiblock techniques are very competitive and can outperform the best layer of the models. Moreover, MFA, UN-PCA and STATIS on BERT-large have no difficulty in achieving the perfect score on UFSAC3 even if the last layers present very poor results separately (cf. Figure 3.2). For RoBERTa-large, multiblock techniques are not as powerful as the best layer, but are still better than the mean of all layers's scores. In general, the performance of MFA is much higher than the average performance of the layers used separately. This indicates that it is very effective for capturing the valuable information

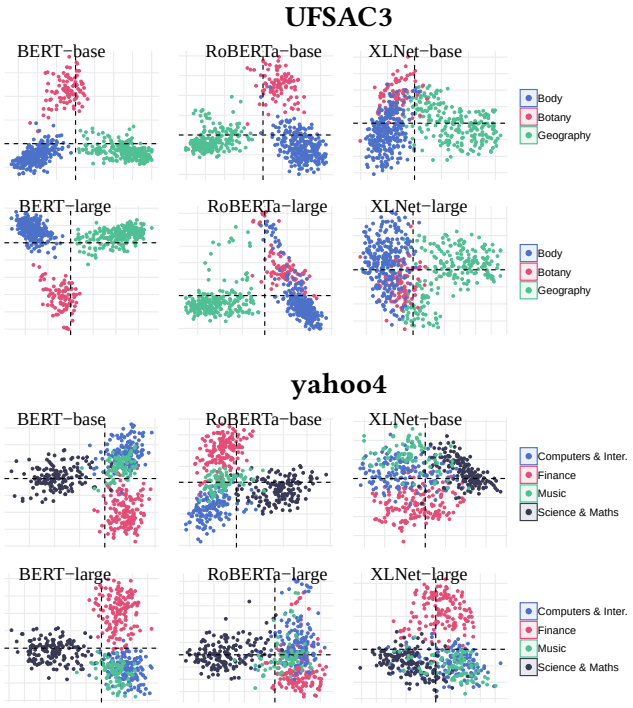

Figure 3.6: 2D projections of UFSAC3 and yahoo4 words on the two first common components of MFA, colored according to the known topics

present in the $d \times b$ features of each model ($9\,216$ for the base models and $24\,576$ for the large ones) in only 15 dimensions.

In Table 3.2 we observe that the raw unfolding (UN) is much less powerful than when reduced with PCA (UN-PCA). This brings out the interest of factor decomposition and dimension reduction in improving performance in the clustering task while combining all layers. This difference in performance is in line with the observations made on Figure 3.5 where the use of PCA noticeably improved the layer-wise clustering, which explains the enhancement observed with UN-PCA.

3.5.3 Visual Interpretation of Factorial Analysis Results

In addition to being one of the best performing multiway techniques, MFA offers several visual interpretation possibilities that are presented in Figures 3.6, 3.7 and 3.8.

Figure 3.6 provides the words' projections on the two first common factors of MFA. Comparing the 6 Transformer models in terms of points' dispersion, we obtain the same ranking of models as made previously, except that RoBERTa-base seems better than RoBERTa-large, which is in line with the numerical results described by MFA's row in Table 3.2.

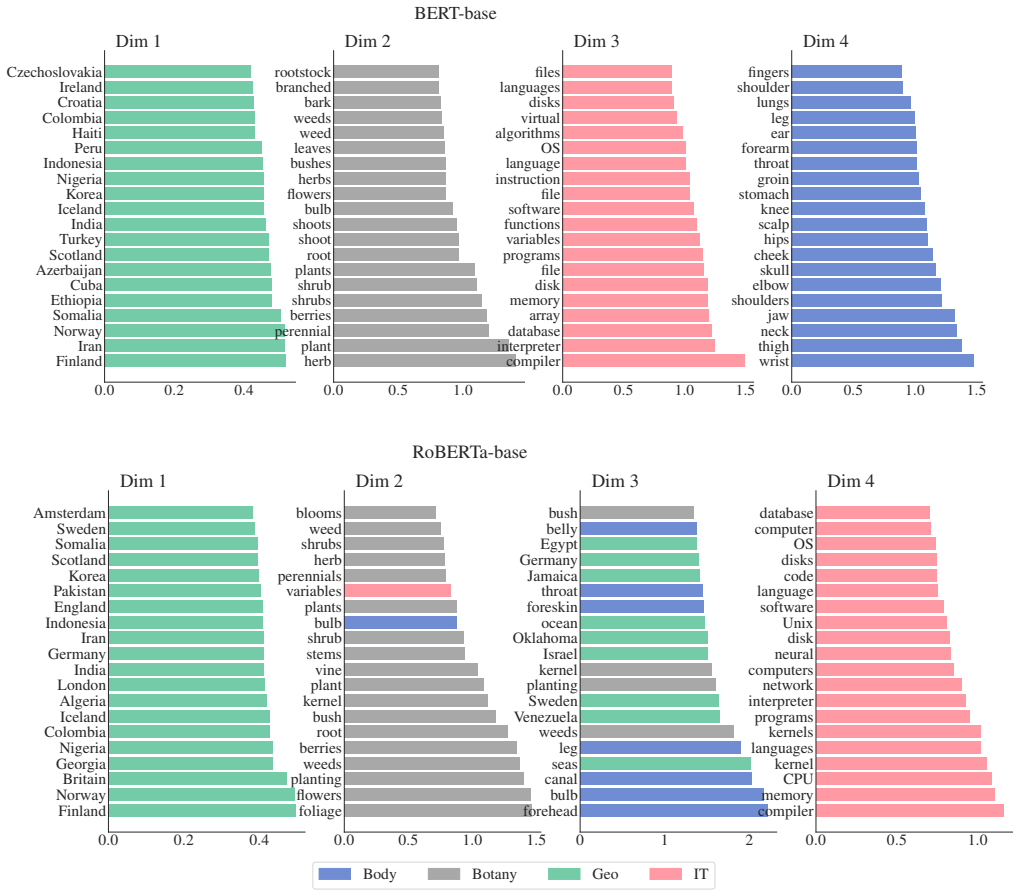

Figure 3.7: The 20 most contributory words (of UFSAC4) to the first three dimensions of MFA applied to all the layers of BERT-base and RoBERTa-base

Figure 3.7 shows the contribution of the first 20 words to the first three components of MFA for BERT-base and RoBERTa-base. We see that for BERT-base, among the 20 most contributory words, the classes are perfectly distributed over the 3 first dimensions (Geography, Botany, IT and Body contribute the most to dimensions 1, 2, 3 and 4 respectively). For RoBERTa-base, the distribution is less homogeneous, the three classes contributing equally to the last dimension.

Figure 3.8 shows the contribution and NMI score of each layer with the 6 models. The contribution of a layer to a component is the sum of the contributions of its variables. More formally, the contribution of the ℓth layer to the jth component is computed as $\sum_{k \in \mathcal{G}_\ell} \boldsymbol{\alpha}_\ell \times \mathbf{u}_{k,j}$ where $\mathbf{u}_{k,j}$ is the loading of the kth variable for the jth component, $\boldsymbol{\alpha}_\ell = \frac{1}{\lambda_1^\ell}$ is the weight of the ℓth layer and $\mathcal{G}_\ell$ is the set of variables of the ℓth layer. The contribution of each variable takes values between 0 and 1 and sums to 1 for a given component. The values of contribution displayed in Figure 3.8 are percentages. The NMI

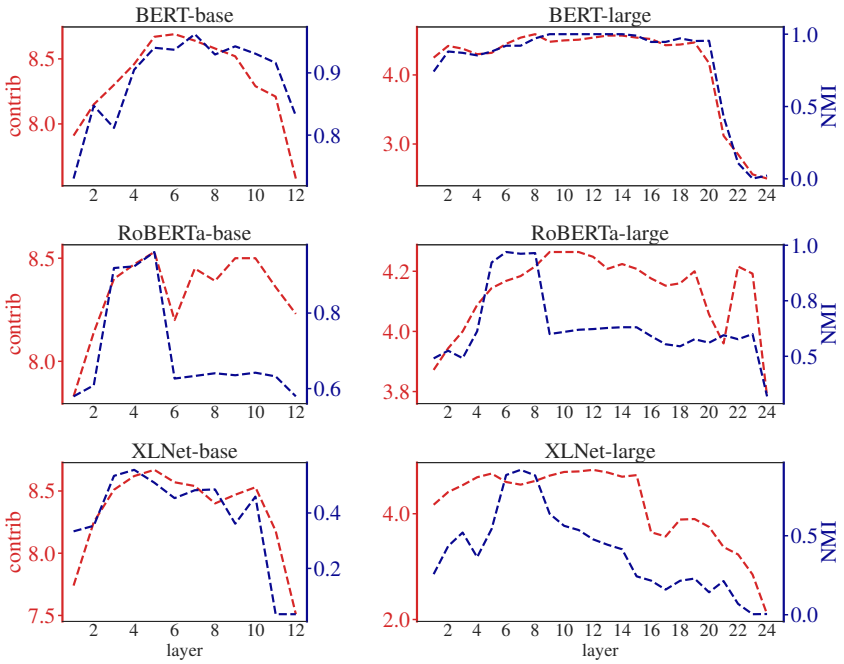

Figure 3.8: The *contribution* of each block (layer) to the first common component of MFA (UFSAC3). The NMI (in red) of a layer ℓ is obtained with clustering separately the $\mathbf{X}_\ell$ matrix made of the original word representations.

metric of each layer ℓ is the same as in Figure 3.2 (UFSAC3). We observe from Figure 3.8 that MFA seems to sum up the relevant information present in the layers of the models, without taking into account the most outlying (and less performing) layers, especially for BERT. For the other models, the drop of performance occurring at the end of some models coincides with a significant decrease of contribution. We can also see that for RoBERTa-base, the best performing layer (layer 5) is also the most contributory to the first component of MFA.

3.6 Conclusion

The present study has shown that a multiway-based approach to word clustering has a twofold benefit. It first removes the need to choose, among the different layers of a Transformer, the one supposed to be the most useful input, something impossible to determine in an unsupervised setting. Second, across a variety of Transformer models as well as datasets for which labels were available for evaluation, multiway techniques have been shown to deliver a performance that most often outperforms that obtained with the most useful layer. Among the multiway techniques, multiblock approaches, especially MFA, stand out as the best performers, outscoring standard tensor-decomposition methods. This

can be attributed to the fact that MFA aims to give a balanced role to the layers. It is based on a factor analysis in which the variables are weighted and these weights are identical for the dimensions of the same layer (and vary from one layer to another). Thereby, we have shown for the first time the interest of using such a method in NLP.

Paths for future research include experimenting with other clustering scenarios, and investigating how to make supervised tasks such as text classification benefit from multiway techniques.

Chapter 4

An Ensemble Approach for Text Clustering with Transformers

Pre-trained Transformer-based word embeddings are now widely used in text mining where they are known to significantly improve supervised tasks such as text classification and named entity recognition and question answering. Since the Transformer models create several different embeddings for the same input, one at each layer of their architecture, various studies have already tried to identify those of these embeddings that most contribute to the success of the above-mentioned tasks. In contrast the same performance analysis has not yet been carried out in the unsupervised setting. In this chapter we evaluate the effectiveness of Transformer models on the important task of text clustering. In particular, we present a clustering ensemble approach that harnesses all the network's layers. Numerical experiments carried out on real datasets with different Transformer models show the effectiveness of the proposed method compared to several baselines.

4.1 Introduction

Starting with BERT (Devlin et al., 2019), Transformer-based contextualized word embeddings provided by neural language models have been increasingly used as the initial input to many NLP applications where they greatly contribute to achieving impressing performance levels.

A Transformer model produces several representations for each word (one at each layer of the network architecture) and studies in the realm of supervised learning have tried to determine the kind of information captured by these different layers. For example, using pre-trained Transformer embeddings as input to a suite of NLP tasks, the authors in (Peters et al., 2018a; Tenney et al., 2019a) have agreed that early layers encode most

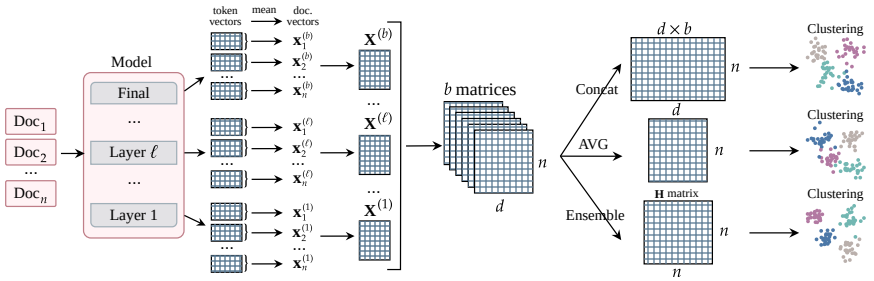

Figure 4.1: Different ways of combining the ℓ layers' embeddings of a Transformer language model. $\mathbf{x}_i^{(\ell)}$ is the document vector computed by averaging the representations (obtained at layer ℓ) of the tokens contained in document i. This vector forms the i-th row of the $\mathbf{X}^{(\ell)}$ matrix, which is the representation of the dataset at layer ℓ.

local syntactic phenomena while more complex semantics appear at the higher layers. This observation has also been made on other languages (de Vries et al., 2020). On the other hand, looking more specifically at the generalization capabilities of contextualized word representations (including ELMo, BERT and OpenAI Transformer algorithms), Liu et al. (2019a) have observed that Transformers' middle layers present a better transferability while Hao et al. (2019) observed that the early layers of BERT-large (24 layers) are more invariant across tasks than the higher layer and hence more transferable. In another line of research, some studies have concentrated on the impact of fine-tuning and have experimentally verified that the closer we get to the last layer, the more task specific the representations are (Kovaleva et al., 2019; van Aken et al., 2019).

The main takeaway of these studies is that embeddings available at the different layers clearly capture different information, thus leading to very different results when used as input to a given text mining task. The problem is that it is not possible to know in advance which one will help to best perform on a given task. When using pre-trained embeddings, a common empirical rule is to exclude the last layer on the assumption that it is biased to the training targets. The very first layers are also commonly excluded since they are deemed to be too close to the original word information. Another approach for selecting a layer to perform a given task is to utilize a labeled dataset as a development set to determine the best layer to use for new datasets (Zhang et al., 2020a). We show that this approach is not optimal in our case, since the best layer is often different from one dataset to another. In addition, we believe that semantic features can be very helpful in text clustering, and it has been observed in (Tenney et al., 2019a) that, unlike syntactic information, which is generally concentrated in a few layers, semantic features are spread out across the entire network. This is why, instead of choosing a unique layer, we prefer leveraging all the

representations provided by Transformer models to perform unsupervised learning. To achieve that, we propose to separately cluster the document representations computed at each layer, then deduce a consensual partition, taking advantage of all the information provided at each level of the deep network. In order to evaluate our approach, we compare it to formerly used baselines including the concatenation and averaging of layers, the use of the second-to-last layer as well as the combination of the four last layers. We also compare our results to those obtained with a standard Bag-Of-Words representation.

Also, we investigate the effect of dimension reduction on the Transformer-based embeddings, a topic which has rarely been studied so far. For a review of this question in the context of word embeddings that predate BERT such as Word2vec, fastText and GloVe, the reader is referred to Raunak et al. (2019) who showed that it was possible to half vector dimensions without significantly altering performance, as well as Mu and Viswanath (2018) who used PCA to remove the dominant components when reconstructing the representation matrix. The present work aims at contributing to fill this gap in the context of Transformer embeddings. We show in Section 3.3 that combining clustering ensemble and dimension reduction allows to significantly increase clustering performance on several real datasets. Section 4.3 then highlights an important advantage of the clustering ensemble, namely the effective estimation of the number of clusters along with an efficient partitioning of data samples, which is very useful when the exact number of clusters is not known.

4.2 Clustering Transformer Embeddings

For a dataset of n documents and a Transformer model with b layers, one obtains b dense representations of size d, one from each layer. Given a layer ℓ, the representation of the i-th document of the dataset is obtained from the embeddings of its 512 first tokens that are pooled together using the mean function as suggested in (Reimers and Gurevych, 2019; Xiao, 2018). This leads to obtain a vector $\mathbf{x}_i^{(\ell)}$ of size d, which constitutes the i-th row of the matrix $\mathbf{X}^{(\ell)}$. The dataset can then be represented by b matrices $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(b)}$ of size $n \times d$ as shown in Figure 4.1. We call a partition the result provided by one of the clustering runs and it contains n labels where the i-th label corresponds to the cluster to which the i-th document is assigned.

For a model with b layers, one can think of running the clustering algorithm on each of the $\mathbf{X}^{(\ell)}$ matrices, and pick the one that yields the "best" result. However, in the unsupervised setting where no labels are available, there is no easy way of knowing which matrix $\mathbf{X}^{(\ell)}$ has given the best result. Hence, we propose and describe two main ways of leveraging the various representations provided by a Transformer model: (1) by aggregating the

$\mathbf{X}^{(\ell)}$, $\ell = 1, \ldots, b$ matrices; (2) by using the $\mathbf{X}^{(\ell)}$ matrices individually as part of an ensemble approach. We also assess the performance obtained when using a PCA-based dimension reduction step, reducing the dimensions to $d' = 100$.

4.2.1 Post-processing of the Transformer Representations

The use of linear dimension reduction is commonplace in NLP and has shown promising improvements on various representations (Mu and Viswanath, 2018; Raunak et al., 2019; Xu et al., 2003). In our study, we investigate the power of principal components in reducing the dimension of dense document representations while preserving the information they contain. In particular, we observed that the use of the first principal components does not alter the performance, even when compressing the original vectors to 10% of the dimensions. More importantly, we show that an additional whitening operation applied on the principal components leads to surprising improvements of the clustering performance while drastically reducing the dimensionality. Considering the definition given in Section 1.3.2, the reduced representation of each vector $\mathbf{x}_i$ is given by $\mathbf{y}_i = \mathbf{x}_i \mathbf{W}_{d'}$. In the whitened version of PCA, it is obtained as follows:

$$y_{ij} = \frac{\mathbf{x}_i \mathbf{w}_j}{\sqrt{\sigma_j}} \quad \forall i = 1, \ldots, n \; ; \; j = 1, \ldots, d'$$

where $\mathbf{w}_j$ is the jth eigen vector of $\mathbf{X}^\top \mathbf{X}$ and σ_j its jth eigenvalue, variance 1. We show that instead of removing the dominant principal components from the reconstruction of $\mathbf{X}$ then recompute the principal components, as proposed in (Mu and Viswanath, 2018; Raunak et al., 2019) for word level tasks, directly normalizing the principal components by their eigenvalues is sufficient to mitigate the impact of the dominant principal components and significantly improve the clustering results.

4.2.2 Prior Aggregation of the Layered Representations

For a document i, the first combination method consists in averaging the b vectors $\mathbf{x}_i^{(\ell)}$, $\ell = 1, \ldots, b$, thus obtaining a unique vector of size d, as experienced in (Vulić et al., 2020). We refer to this method as AVG. The second method, which we call Concat, consists in concatenating the b vectors $\mathbf{x}_i^{(\ell)}$, which results in a unique vector of size $b \times d$, as performed in (Devlin et al., 2019) using the last layers. On top of these aggregated representations, we optionally perform a PCA-based dimension reduction before running the clustering algorithm, obtaining representations of size $d' = 100$.

4.2.3 Clustering Ensemble Approach

Another way of combining the information provided by all layers, is not to physically aggregate them, but to rely on a consensus procedure. This clustering ensemble or consensus is referred to as ENS. Ensemble learning has been considered in different machine learning contexts where it generally helps in improving results by combining several models (Affeldt et al., 2020a; Dietterich, 2000). The ensemble approach allows a better predictive performance and a more robust clustering as compared to the results obtained with a single model (Berikov and Pestunov, 2017; Strehl and Ghosh, 2002).

Algorithm 3: Clustering Ensemble

input : A dataset $\mathcal{D}$; a Transformer model $\mathcal{M}$ of b layers, two clustering algorithms $\mathcal{C}_1$ and $\mathcal{C}_2$; the number of clusters g
output : A consensual clustering partition $\mathbf{p}^*$

foreach $\ell = 1, \ldots, b$ **do**
  $\mathbf{X}^{(\ell)} \leftarrow$ document embeddings computed using $\mathcal{M}(\mathcal{D})$ at the ℓ-th layer (as shown in Figure 4.1);
  $\mathbf{p}^{(\ell)} \leftarrow \mathcal{C}_1(\mathbf{X}^{(\ell)}, k)$;
end
$\mathbf{H} \leftarrow$ the association matrix of the $\mathbf{p}^{(\ell)}$ partitions;
$\mathbf{H}^r \leftarrow$ The null-model association matrix from random permutations of the partitions $\mathbf{p}^{(\ell)}$;
$\tau = average(\mathbf{H}^r)$;
foreach i, j *from* 1 *to* n **do**
  if $h_{ij} < \tau$ **then**
    $h_{ij} \leftarrow 0$;
  end
end
$\mathbf{p}^* \leftarrow \mathcal{C}_2(\mathbf{H})$;
return $\mathbf{p}^*$;

Vega-Pons and Ruiz-Shulcloper (2011) identified two main steps in the clustering ensemble process: the generation step and the consensus step. The generation mechanism is responsible for creating the set of partitions and can be carried out in several way such as using different clustering algorithms (Strehl and Ghosh, 2002), different solution initializations (Alizadeh et al., 2014), different sets of hyperparameters (Jia et al., 2011) and different data representations (Tsai et al., 2014). Our approach belongs to the last category as we use different vector representations of the same documents, given as input to the same model, with the same hyperparameters. In Section 6.3.1 we use the same representation and the same algorithm but vary the number of classes. This can be considered as belonging to the

"different hyperparameter" approach since the number of classes can be considered as a hyperparameter. The approach we propose in Section 4.3 lies at the intersection between the two categories since we vary both the input representations and the number of clusters.

Ensemble approaches can also be categorized according to the consensus function they make use of. One common approach is to perform a majority vote after relabeling the cluster partitions so that the pairwise correspondence of the groups is maximized. The consensus vote can be achieved with different strategies adapted to both hard (Fred, 2001) and fuzzy (Dimitriadou et al., 2002) clustering partitions. Another approach is to rely on a bipartite graph that connects the n samples and the g clusters using an association matrix of size $n \times g$ (Fern and Brodley, 2004). Alternatively, Jafarzadegan et al. (2019) used PCA to combine several hierarchical clustering partitions represented by descriptor matrices, thus considering the relation between each element when computing the ensemble partition.

In our present work, following the ensemble paradigm, we use the association matrix $\mathbf{H}_{n\times n} = (h_{ij})$ (called similarity matrix in Strehl and Ghosh 2002) to compute the consensus partition as described in Algorithm 3, where the clustering algorithm $\mathcal{C}_1$ is used on the $\mathbf{X}_\ell$ matrices to obtain the b partitions, while $\mathcal{C}_2$ is used on $\mathbf{H}$ to compute the consensus partition $\mathbf{p}^*$. h_{ij} denotes the number of partitions within $\mathbf{p}^{(\ell)}, \ell = 1, \ldots, b$ that assign the individuals i and j to the same cluster. In order to leverage $\mathbf{H}$, we propose to use a simplified and faster version of the approach proposed in (Bassett et al., 2013). Note that $\mathbf{H}$ can be assimilated to a graph adjacency matrix. To cluster the $\mathbf{H}$ matrix, we use as the parameter $\mathcal{C}_2$ a clustering algorithm that doesn't necessarily require to set the number of clusters in advance. In our experiments we used the Louvain algorithm (Blondel et al., 2008) to obtain the consensus partition, which worked better than K-means on $\mathbf{H}$. In step 7 of Algorithm 3, we use the $mean$ of the *null association* matrix $\mathbf{H}^r$ instead of the max used in (Bassett et al., 2013). The reason is that in our case, the number of partitions (equal to the number of layers b) is relatively small. This conducts the largest value of the randomly shuffled association matrix to tend easily to the largest possible value (i.e. the number of partitions b).

We use the following example to illustrate the ensemble approach, in which the partitions $\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \mathbf{p}^{(3)}$ correspond to the $\mathbf{p}^{(\ell)}$ partitions:

$$\begin{array}{ccc} \mathbf{p}^{(1)} & \mathbf{p}^{(2)} & \mathbf{p}^{(3)} \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 2 & 1 & 2 \\ 2 & 1 & 2 \\ 3 & 1 & 2 \end{array} \xrightarrow[\text{matrix}]{\text{agreement}} \mathbf{H} = \begin{pmatrix} 3 & 2 & 3 & 0 & 0 & 0 \\ 2 & 3 & 2 & 0 & 0 & 0 \\ 3 & 2 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 3 & 2 \\ 0 & 0 & 0 & 3 & 3 & 2 \\ 0 & 0 & 0 & 2 & 2 & 3 \end{pmatrix} \xrightarrow[\text{maximization}]{\text{modularity}} \begin{array}{c} \mathbf{p}^* \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{array}$$

4.2.4 Experimental Study and Compared Results

In the clustering experiments, we use as $\mathcal{C}_1$ in Algorithm 3 the best of 10 runs (in terms of inertia) of k-means and as $\mathcal{C}_2$ the Louvain Algorithm. To validate the results produced by the clustering we rely on standard external measures devoted to assessing cluster quality namely Normalized Mutual Information (NMI) (Strehl and Ghosh, 2002) and Adjusted Rand Index (ARI) (Hubert and Arabie, 1985; Steinley, 2004).

Datasets and Models Used

The datasets used for clustering experiments are described in Table 4.1, where the balance is the ratio between the smallest and largest cluster sizes. We used classic3 and classic4 datasets of Cornell University, the BBC news dataset proposed in (Greene and Cunningham, 2006) and random extracts of DBPedia (Lehmann et al., 2015) and AG-news[1] of size 12,000 and 8,000 respectively. From each set of documents, we compute multiple contextual representations from 4 pre-trained models which are the base (12 layers) and large (24 layers) versions of BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019b), with a vocabulary size of 28,996 for BERT and 50,265 for RoBERTa.

Table 4.1: Datasets' description.

	classic3	classic4	DBPedia	AG-news	BBC
Clusters	3	4	14	4	5
Balance	0.71	0.32	0.92	0.97	0.76
Samples	3,891	7,095	12,000	8,000	2,225

Layer-wise Clustering Results

Layer-wise clustering results are presented in Figure 4.2 for two datasets. The Figure shows that reducing the number of dimensions to only 100 (which constitutes less than 10% of the large model's features) leads to a significant improvement of performance, especially in the case of RoBERTa-base, for which we observe an increase of at least 0.54 in NMI score for all layers on the classic3 dataset. We can also observe that, from a dataset to another, the best layer is not always the same. Indeed, if we take the example of BERT-base, the best layers for the 5 datasets are respectively the 1-st, 11-th, 9-th, 2-nd and 1-st. Moreover, we sometimes observe several layers presenting good results, which indicates that all layers

[1] http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

can bring useful information, potentially different from one to another as discussed in (Peters et al., 2018a; Tenney et al., 2019a; Vulić et al., 2020).

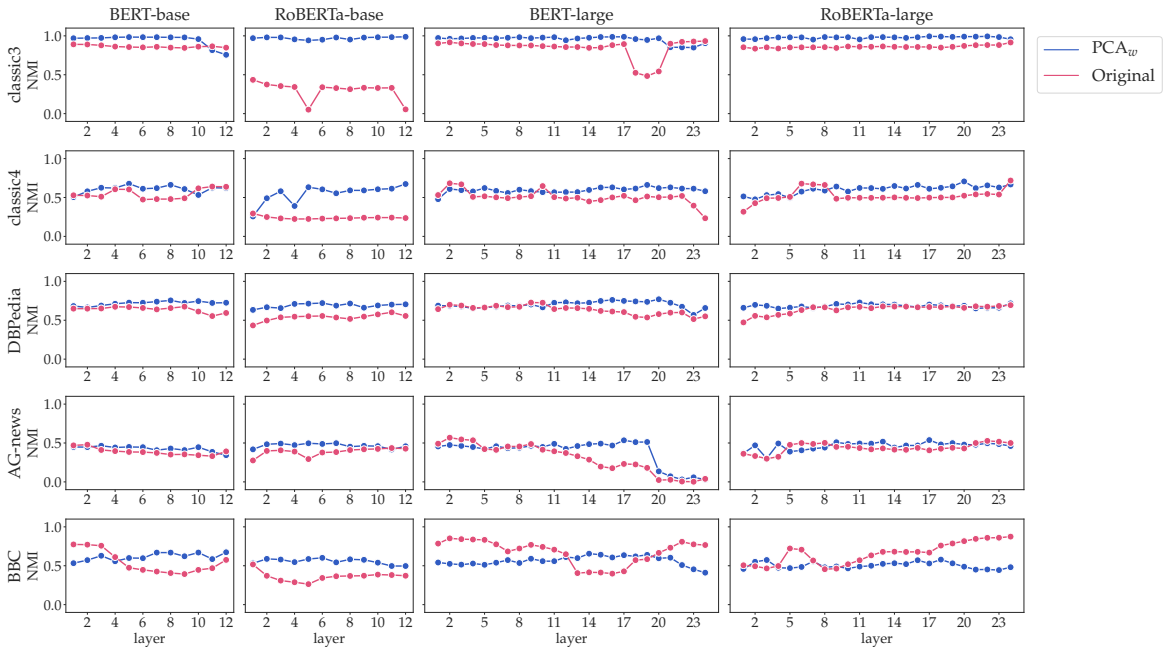

Figure 4.2: Clustering performance (NMI) across layers of the original representations of pre-trained models (using all of the d dimensions of the $\mathbf{X}^{(\ell)}$ matrices) compared to reduced representations ($d' = 100$), with $\ell = 1, \ldots, b$.

Multi-layer Clustering Results

Since the obtained results may greatly vary with each layer, as clearly visible in Figure 4.2, and since determining which one is the best is very difficult in the absence of labels, we propose to simultaneously use all the $\mathbf{X}^{(\ell)}$ data matrices $\ell = 1, \ldots, b$ provided by the network as describes in Sections 4.2.2 and 4.2.3. Table 4.2 presents the NMI obtained by each technique assuming that the number of clusters g is known.

We compare the all-layered approach to several baselines. The first one is the use of a Bag-Of-Words (BOW) representation as input to a Spherical K-means (Dhillon and Modha, 2001) algorithm, known to be well suited to directional sparse data compared to K-means. Two other baselines are the use of the second-to-last layer (Devlin et al., 2019; Xiao, 2018) as well as the combination of the four last layers (Devlin et al., 2019). The *mean rank* corresponds to the average of all the ranks assigned to a given method based on its performance compared to the other methods, over all the model-dataset combinations.

A first observation is the effectiveness of the PCA-based dimension reduction when comparing the scores obtained with raw vectors and their reduced version (e.g. for AVG,

Table 4.2: NMI scores of multi-layer clustering techniques over the four Transformer models on document clustering. LW (layer-wise) corresponds to the mean of the NMI scores obtained by each layer ℓ (using $\mathbf{X}^{(\ell)}$) and the value between brackets to the score obtained by the best layer, a layer that unfortunately can't be identified in the absence of labels. Note that the lower the *meank rank*, the better.

Dataset/		BOW	LW mean (best)		2nd-to-last		4 last layers (raw)			All layers (raw)			4 last (PCA_w)			All (PCA_w)		
	Model		Raw	PCA_w	Raw	PCA_w	AVG	UN	ENS	AVG	UN	ENS	AVG	UN	ENS	AVG	UN	ENS
classic3	$BERT_b$	0.95	0.86 (0.89)	0.94 (0.98)	0.87	0.82	0.85	0.87	0.87	0.85	0.87	0.88	0.95	0.78	**0.98**	**0.98**	0.92	**0.98**
	$BERT_l$		0.84 (0.93)	0.95 (0.99)	0.93	0.85	0.93	0.92	0.93	0.9	0.91	0.9	0.84	0.82	0.98	0.98	0.95	**0.99**
	$RoBERTa_b$		0.3 (0.43)	0.97 (0.99)	0.33	**0.98**	0.06	0.33	0.33	0.06	0.33	0.34	**0.98**	0.94	**0.98**	0.94	0.95	**0.98**
	$RoBERTa_l$		0.86 (0.91)	0.98 (0.99)	0.88	0.98	0.89	0.89	0.89	0.86	0.86	0.86	**0.99**	0.96	**0.99**	0.97	0.97	**0.99**
classic4	$BERT_b$	0.64	0.55 (0.64)	0.61 (0.68)	0.64	0.63	0.62	0.64	0.64	0.61	0.63	0.53	0.61	0.59	**0.74**	0.63	0.56	0.73
	$BERT_l$		0.51 (0.68)	0.59 (0.66)	0.4	0.61	0.52	0.54	0.58	0.68	0.53	0.53	0.56	0.57	0.64	0.66	0.6	**0.74**
	$RoBERTa_b$		0.24 (0.29)	0.55 (0.67)	0.24	0.61	0.24	0.24	0.24	0.23	0.24	0.25	0.59	0.65	0.7	0.58	0.55	**0.74**
	$RoBERTa_l$		0.52 (0.72)	0.6 (0.71)	0.54	0.63	0.55	0.55	0.54	0.51	0.52	0.54	0.67	0.61	**0.75**	0.69	0.67	**0.75**
DBPedia	$BERT_b$	0.69	0.64 (0.67)	0.72 (0.76)	0.55	0.72	0.65	0.61	0.57	0.69	0.67	0.69	**0.75**	0.74	0.71	0.74	0.72	**0.75**
	$BERT_l$		0.63 (0.73)	0.7 (0.77)	0.51	0.57	0.61	0.59	0.61	0.68	0.65	0.73	0.67	0.67	0.62	0.71	0.71	**0.76**
	$RoBERTa_b$		0.54 (0.6)	0.69 (0.72)	0.6	0.7	0.57	0.58	0.54	0.55	0.56	0.49	**0.73**	0.69	0.56	0.72	0.69	0.7
	$RoBERTa_l$		0.64 (0.69)	0.68 (0.73)	0.68	0.66	0.69	0.68	0.68	0.69	0.66	0.68	0.67	0.7	0.58	0.66	0.7	**0.73**
AG-news	$BERT_b$	0.46	0.39 (0.48)	0.43 (0.46)	0.33	0.39	0.4	0.37	0.39	0.44	0.42	0.41	0.43	0.44	0.36	0.43	0.36	**0.54**
	$BERT_l$		0.3 (0.57)	0.38 (0.53)	0.0	0.06	0.17	0.01	0.0	0.49	0.21	0.49	0.11	0.03	0.0	0.5	0.2	**0.54**
	$RoBERTa_b$		0.39 (0.44)	0.47 (0.5)	0.44	0.42	0.43	0.44	0.43	0.41	0.42	0.41	0.46	0.47	0.47	0.48	0.44	**0.58**
	$RoBERTa_l$		0.44 (0.53)	0.46 (0.54)	0.52	0.49	0.53	0.53	0.52	0.51	0.49	0.46	0.46	0.46	0.53	0.53	0.47	**0.59**
BBC	$BERT_b$	0.75	0.55 (0.77)	0.61 (0.67)	0.47	0.59	0.46	0.45	0.45	0.6	0.51	0.55	0.63	0.66	0.74	0.54	0.61	**0.8**
	$BERT_l$		0.67 (0.85)	0.57 (0.66)	0.78	0.45	0.82	0.82	0.79	0.79	0.78	0.79	0.43	0.4	0.62	0.53	0.5	**0.88**
	$RoBERTa_b$		0.36 (0.52)	0.56 (0.6)	0.38	0.5	0.37	0.38	0.39	0.34	0.38	0.38	0.53	0.54	0.64	0.62	0.48	**0.83**
	$RoBERTa_l$		0.66 (0.87)	0.5 (0.58)	**0.86**	0.44	**0.86**	**0.86**	**0.86**	0.74	0.74	0.74	0.52	0.5	0.65	0.51	0.57	0.79
Mean rank		5.58	-	-	10.35	9.05	9.55	9.35	9.75	9.68	10.45	9.88	7.32	8.32	6.10	5.58	7.45	**1.60**

Concat and ENS using all layers, we move from a *mean rank* of 9.68, 10.45 et 9.88 to 5.58, 7.45 et 1.6 resp.). The results also show that the use of the second-to-last layer is not reliable, as its effectiveness highly depends on the model and the dataset. Combining the four last layers is more effective but presents lower performance than the use of the all of the layers. This suggests that the useful information provided by the embeddings is different from one layer to another. Also, all of the useful information seems to be efficiently captured by the reduced dimensions (see the Concat results, for which we move from $d \in \{9\,216, 24\,576\}$ to only 100). Moreover, our results clearly show a significant advantage of the ensemble technique over the other approaches of combining layers, presenting the highest results in terms of NMI and the lowest *mean rank* by far. This is further confirmed in Table 4.3 by two other metrics (ARI and Accuracy) that both place our ensemble approach in the first position.

Table 4.3: Mean rank values according to three different performance metrics.

Metric	BOW	2nd-to-last		4 last layers (raw)			All layers (raw)			4 last (PCA_w)			All (PCA_w)		
		Raw	PCA_w	AVG	UN	ENS	AVG	UN	ENS	AVG	UN	ENS	AVG	UN	ENS
NMI	5.58	10.35	9.05	9.55	9.35	9.75	9.68	10.45	9.88	7.32	8.32	6.1	5.58	7.45	**1.60**
ARI	3.98	9.75	9.65	8.85	9.12	8.9	9.02	9.72	10.15	7.75	8.95	6.75	6.32	8.32	**2.75**
ACC	5.03	9.82	9.0	8.7	8.88	9.52	8.12	9.22	9.78	7.35	8.55	7.45	6.22	8.38	**3.98**

4.3 Clustering with an Estimated Number of Clusters

All the results presented previously are based on the assumption that the exact number of clusters is known in advance, which is not realistic in real life. In contrast, another significant advantage offered by the proposed approach is the use of an algorithm for which the number of clusters is not known *a priori* (Louvain in our experiments). This means that the consensus returns a partition with clusters that respect the original cluster assignments as much as possible, without necessarily providing the same number of clusters as the input partitions. It is therefore a good alternative when the exact number of clusters is unknown. Also, the number of clusters for each partition is not necessarily the same, which is an interesting feature of the ensemble clustering. In order to benefit from this property, given a set $\mathcal{G}$ of some selected values of g, we ensure that each K-means run takes as input a value $g \in \mathcal{G}$ while covering as much as possible the whole set of values. We use in our experiments $\mathcal{G} = [g_r - 5, g_r + 5]$ where g_r is the real number of clusters, e.g. for the DBPedia dataset where $g_r = 14$ using a "base" model with 12 layers, the list of the 12 values of g could possibly be $\{9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 9\}$, obtaining 12 partitions $\mathbf{p}^{(1)}, \ldots \mathbf{p}^{(\ell)}, \ldots \mathbf{p}^{(12)}$ from which we compute the consensual partition $\mathbf{p}^*$ as described in Algorithm 3. The $\mathbf{p}^*$ partition then groups in the same cluster the individuals that are usually grouped together in the input partitions without having the constraint of a fixed g. This guarantees a robust clustering performance while automatically estimating the number of clusters. Table 4.4 shows the results of the ensemble algorithm (using the compressed representations) obtained with varying $g \in [g_r - 5, g_r + 5]$ (g_r is given in Table 4.1).

We observe from Table 4.4 that the performance is not significantly altered, even when the estimated number of clusters is not equal to g_r. For classic3, classic4 and AG-news, the clustering ensemble with varying g always finds g_r clusters, except for classic4 using RoBERTa-base, where a partition of 5 clusters is found but with a high NMI, which is explained by the fact that the extra cluster corresponds to the split of the "cacm" class (cf. Figure 4.3). On the contrary, the number of clusters is underestimated for the BBC dataset (4 clusters instead of 5), for which the classes "tech" and "entertainement" seem to

Table 4.4: Performance obtained by clustering ensemble using all the layers with an estimated number of clusters.

Model	classic3			classic4			DBPedia			AG-news			BBC		
	$\hat{k}$	NMI	ARI	$\hat{k}$	NMI	ARI	$\hat{k}$	NMI	ARI	$\hat{k}$	NMI	ARI	$\hat{k}$	NMI	ARI
BERT-base	3	0.98	0.99	4	0.74	0.52	8	0.71	0.49	4	0.55	0.56	4	0.69	0.62
BERT-large	3	0.98	0.99	4	0.73	0.52	9	0.77	0.54	4	0.5	0.45	4	0.74	0.64
RoBERTa-base	3	0.98	0.99	5	0.79	0.65	10	0.78	0.59	4	0.52	0.49	4	0.74	0.65
RoBERTa-large	3	0.98	0.99	4	0.74	0.53	7	0.68	0.41	4	0.59	0.51	4	0.75	0.65

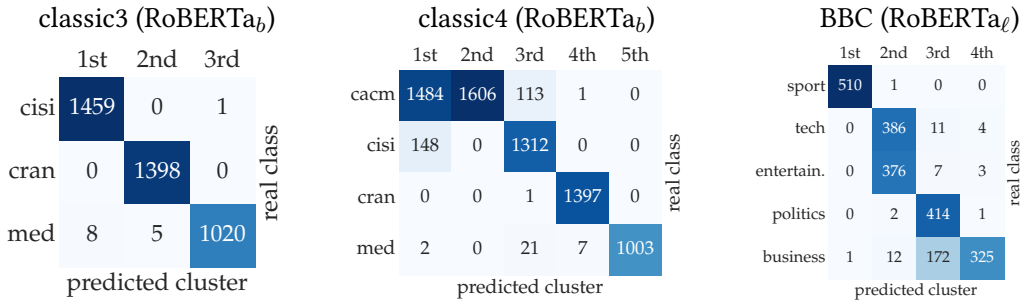

Figure 4.3: Confusion matrices obtained by clustering with an estimated number of clusters (as described in Section 4.3).

be merged, as well as the "politics" class and some "business" examples. For DBPedia, the representations provided by RoBERTa-base are the ones presenting an estimation of g that is the closest to g_r, along with a high NMI score. In that case, as for BBC, some classes are merged together such as "animal" with "plant" and "film" with "written work".

4.4 Comparison with End-to-End Approaches

In this section, we compare our results with multi-view techniques that concomitantly learn from several numerical representations of the same objects.

Graph-based Approach

Tensor Graph Modularity (TGM) (Boutalbi et al., 2022) is a graph-based multi-view technique that relies on a three-way version of the modularity function. The objective function of TGM is the generalization of the two-way modularity (Ailem et al., 2015; Labiod and Nadif, 2011) to tensor data. The modularity is commonly used to estimate the quality of graph clustering by comparing the edge density belonging to a cluster (or community) against the expected density of a random graph.

Given v graphs represented by v adjacency matrices $\mathbf{A}^{(\ell)},\ \ell = 1, \ldots, v$ of size $n \times n$ and each containing n nodes and $e^{(\ell)}$ edges, the aim of TGM is to find the partition matrix $\mathbf{Z}_{n\times k} = (z_{ig})$ that minimizes the three-way modularity, where $(z_{ig}) = 1$ if the ith document is affected to the gth cluster and 0 otherwise. The objective function of TGM can be written as:

$$\underset{\mathbf{Z}}{\text{minimize}} \sum_{\ell=1}^{v} \frac{1}{a_{\cdot\cdot}^{(\ell)}} \text{Trace}[(\mathbf{A}^{(\ell)}\mathbf{Z} - \mathbf{M}^{(\ell)}\mathbf{Z})\mathbf{Z}^{\top}] \tag{4.1}$$

where $\mathbf{M}^{(\ell)} = (m_{ij}^{(\ell)}) = (\frac{a_{i\cdot}^{(\ell)} a_{\cdot j}^{(\ell)}}{2e^{(\ell)}})$ and represents the expected probabilities of ℓth graph edges. For more detail about the optimisation of TGM, the reader is referred to our recently published paper (Boutalbi et al., 2022).

The idea behind TGM is to leverage v different representations of a corpus $\mathcal{D}$, including sparse (BOW and entity linking) and dense vector representations (GloVe, BERT and SRoBERTa). Each vectorization technique provides a representation $\mathbf{X}^{(\ell)}$ from which a graph adjacency matrix $\mathbf{A}^{(\ell)}$ is computed. In our particular case, we use the $v = b$ representations provided by a given Transformer model (cf. step 2 in Algorithm 3) to construct the adjacency matrices $\mathbf{A}^{(\ell)},\ \ell = 1, \ldots, b$. Originally, the adjacency matrices we derive from sparse representations contain the number of words or entities shared by each pair of documents. In the case of dense representations, we compute a pair-wise similarity matrix computed as follows:

$$a_{ij}^{(\ell)} = \begin{cases} 1 & \text{if } f(\mathbf{x}_i^{(\ell)}, \mathbf{x}_j^{(\ell)}) \geq p \\ 0 & \text{otherwise} \end{cases},$$

where $\mathbf{x}_i^{(\ell)}$ and $\mathbf{x}_j^{(\ell)}$ are the ith and jth row of $\mathbf{X}^{(\ell)}$ and correspond to the ith and jth document representations at the bth layer. f is the similarity measure used to assess the proximity between two vector representations $\mathbf{x}_i^{(\ell)}$ and $\mathbf{x}_j^{(\ell)}$. p is the percentile that depends on the desired sparsity of the similarity matrix. In our experiments, we use the percentile that leads to a sparsity of 97%. The same process is used for the b layers, thus obtaining $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(b)}$ adjacency matrices.

Multi-view Linear Approach

We also consider a multi-view version of simultaneous dimension reduction and clustering proposed in (Fettal et al., 2023) for graph clustering but that can be generalized to other data types. We investigate this approach for it may be considered as a multi-view version of simultaneous clustering (via the k-means criterion) and linear embedding (De Soete and Carroll, 1994), which makes it closely related to our ensemble approach.

In order to adapt LMGEC (Linear Multi-view Graph Embedding and Clustering) to our work on Transformers, we make a few adjustments. First, to make it suitable to our work on Transformers, we skip the graph-specific pre-processing steps and use each of our $\mathbf{X}^{(\ell)}$ matrices as a feature matrix, used as input into the algorithm. We hence use only the part of the algorithm that minimizes the criterion:

$$\sum_{\ell=1}^{b} \alpha_\ell \|\mathbf{X}^{(\ell)} - \mathbf{Z}\mathbf{F}\mathbf{W}^{(\ell)\top}\|_F^2 \tag{4.2}$$

where $\mathbf{W}^{(\ell)}$ can be associated to the encoding or embedding matrix of size $d \times d'$, $\mathbf{F}$ is the $g \times d$ matrix containing the centroids and $\mathbf{Z}$ the membership matrix of size $n \times g$. In real terms, the objective is to find the partition (parameters $\mathbf{Z}$ and $\mathbf{F}$) as well as the subspaces of size d' (parameters $\mathbf{W}^{(\ell)},\ \ell = 1, \ldots, b$) so that the reconstruction of the centroids $\mathbf{Z}\mathbf{F}\mathbf{W}^{(\ell)\top}$ is as close as possible to the original data $\mathbf{X}^{(\ell)},\ \forall \ell = 1, \ldots, b$. For more detail about the optimization process, please refer to Fettal et al. (2023).

Whitening. We add a whitening step to the optimization process in order to make it more suitable for our data and improve the results. The whitening step is injected in two ways:

- Initializing $\mathbf{W}^{(\ell)}$: given $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \texttt{TuncatedSVD}(\mathbf{X}^{(\ell)})$, $\mathbf{W}^{(\ell)}$ is initialized as $\mathbf{V}/\mathbf{\Sigma},\ \forall \ell = 1, \ldots, b$.
- Updating $\mathbf{W}^{(\ell)}$: at each iteration, each column $\mathbf{w}_j^{(\ell)}$ of $\mathbf{W}^{(\ell)} = [\mathbf{w}_1^{(\ell)}, \ldots, \mathbf{w}_j^{(\ell)}, \ldots, \mathbf{w}_{d'}^{(\ell)}]$ is computed as:

$$\mathbf{w}_j^{(\ell)} = \frac{\mathbf{U}\mathbf{v}_j}{\sqrt{\sigma^2(\mathbf{X}^{(\ell)}\mathbf{U}\mathbf{v}_j)}};\ j = 1, \ldots, d'; \ell = 1, \ldots, b$$

  where $\mathbf{U}$ and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_j, \ldots, \mathbf{v}_{d'}]^T$ contain respectively the left and right singular vectors of $(\mathbf{X}^{(\ell)}\mathbf{Z}\mathbf{F})$ and $\sigma^2(\mathbf{y}) = \frac{\sum_{i=1}^{n}(y_i - \bar{\mathbf{y}})^2}{n}$ is the variance of the vector $\mathbf{y}$ of size n and mean $\bar{\mathbf{y}}$.

These adjustments lead to an end-to-end version of our ensemble approach and is referred to as LMGEC$_w$. Both whitening steps have proven effective and significantly improve the clustering results of the algorithm (cf. Figure B.2 in the Appendix).

Dataset	Model	NMI			ARI			Accuracy		
		TGM_w	$LMGEC_w$	ENS_w	TGM_w	$LMGEC_w$	ENS_w	TGM_w	$LMGEC_w$	ENS_w
classic3	BERT-base	0.936	0.878	**0.983**	0.962	0.914	**0.992**	0.987	0.971	**0.997**
	BERT-large	0.966	0.907	**0.987**	0.983	0.938	**0.994**	0.994	0.979	**0.998**
	RoBERTa-base	0.945	0.060	**0.985**	0.969	0.060	**0.993**	0.989	0.435	**0.997**
	RoBERTa-large	0.96	0.871	**0.989**	0.979	0.908	**0.995**	0.993	0.968	**0.998**
classic4	BERT-base	**0.854**	0.523	0.734	**0.873**	0.320	0.525	**0.954**	0.516	0.771
	BERT-large	**0.875**	0.531	0.737	**0.889**	0.327	0.526	**0.96**	0.519	0.771
	RoBERTa-base	**0.766**	0.404	0.744	**0.764**	0.306	0.529	**0.91**	0.553	0.772
	RoBERTa-large	**0.78**	0.526	0.748	**0.771**	0.322	0.532	**0.914**	0.519	0.774
DBPedia	BERT-base	0.672	0.666	**0.751**	0.503	0.487	**0.51**	**0.619**	0.594	0.543
	BERT-large	-	0.709	**0.759**	-	0.517	**0.537**	-	**0.603**	0.543
	RoBERTa-base	0.642	0.557	**0.699**	**0.504**	0.327	0.426	**0.607**	0.454	0.442
	RoBERTa-large	-	0.639	**0.728**	-	0.426	**0.478**	-	0.502	**0.52**
AG-news	BERT-base	0.491	0.391	**0.543**	0.488	0.329	**0.545**	0.751	0.583	**0.797**
	BERT-large	0.488	0.448	**0.54**	0.485	0.414	**0.57**	0.744	0.716	**0.802**
	RoBERTa-base	0.379	0.434	**0.576**	0.342	0.408	**0.53**	0.565	0.721	**0.754**
	RoBERTa-large	0.456	0.497	**0.585**	0.445	0.465	**0.531**	0.704	**0.752**	0.732
BBC	BERT-base	0.597	0.583	**0.802**	0.526	0.524	**0.758**	0.715	0.761	**0.882**
	BERT-large	0.688	0.675	**0.877**	0.59	0.580	**0.889**	0.697	0.691	**0.953**
	RoBERTa-base	0.741	0.393	**0.827**	0.744	0.294	**0.84**	0.889	0.545	**0.931**
	RoBERTa-large	0.671	0.546	**0.794**	0.654	0.444	**0.757**	0.836	0.611	**0.879**

Table 4.5: Clustering results according to NMI, ARI and Accuracy on the five datasets with four models. Missing values are due to out of memory errors.

Compared Results

The obtained clustering results are depicted in Table 4.5 with 10 different initializations. We show the results of the whitened version of TGM, LMGEC and ENS for they provide the best results compared to the basic version. TGM_w means that we use the PCA_w post-processing before we compute the similarity matrices $\mathbf{A}^{(\ell)},\ \ell = 1, \ldots, b$. For $LMGEC_w$ we set the temperature $\tau = 10$ as recommended in (Fettal et al., 2023), which gives a greater weight to high-inertia layers (in the sense of the SSE criterion).

Overall, the two end-to-end techniques achieve competitive performance scores, especially TGM_w which outperforms ENS_w and $LMGEC_w$ on classic4 with all of the models. $LMGEC_w$ seems to be the approach that least benefits from the whitening operation. Indeed, even though the results are improved, it still struggles with RoBERTa-base, that provides representations of poor quality, unless they're post-processed using PCA_w (as shown in Figure 3.2). TGM_w does not seem impacted by the poor results of RoBERTa-base in contrast to when TGM is used without PCA_w. The ensemble approach remains the most competitive, especially on classic3 and BBC.

4.5 Results on Word Clustering Datasets

Table 4.6 presents the NMI obtained by each technique assuming that the number of clusters g us known. We observe from Table 4.6 that in the case of BERT (which is the best model in both the base and large version), combining all the layers can perform better than the best layer of the same model. Moreover, ENS and Concat with BERT-large have no difficulty in achieving the perfect score on UFSAC3 even if the last layers present very poor results separately (Figure 3.2). For RoBERTa, the consensus is not as powerful as the best layer, but is still better than the mean of all layers's scores. Multi-layer clustering using the PCA-reduced version of word embeddings can often be even more competitive, e.g. for RoBERTa-base on UFSAC3, applying PCA_w allows to move from NMI scores of 0.94, 0.72 and 0.94 to 0.96, 0.96 and 0.97 for AVG, ENS and Concat respectively.

Table 4.6: Compared performance (NMI scores) of over the four models and fastText. For "Raw" and "PCA_w" the scores correspond to averages over all the layers (except of course for fastText). The value between brackets corresponds to the score obtained by the best layer, a layer that unfortunately can't be identified in the absence of labels.

Dataset	Model	fastText		LW mean (best)		AVG		UN		ENS	
		Raw	PCA_w	Raw	PCA_w	Raw	PCA_w	Raw	PCA_w	Raw	PCA_w
UFSAC3	BERT-base	0.92	0.92	0.89 (0.96)	0.91 (0.98)	0.95	0.98	0.96	0.98	0.97	0.98
	BERT-large			0.81 (1.00)	0.80 (1.00)	0.98	0.99	1.00	1.00	1.00	1.00
	RoBERTa-base			0.87 (0.95)	0.90 (0.97)	0.94	0.96	0.94	0.97	0.95	0.97
	RoBERTa-large			0.86 (0.98)	0.88 (0.98)	0.76	0.79	0.66	0.65	0.95	0.96
UFSAC4	BERT-base	0.77	0.84	0.86 (0.94)	0.88 (0.95)	0.90	0.96	0.96	0.97	0.95	0.96
	BERT-large			0.72 (0.98)	0.75 (0.96)	0.93	0.97	0.97	0.99	0.99	0.99
	RoBERTa-base			0.78 (0.86)	0.84 (0.91)	0.87	0.91	0.67	0.76	0.88	0.91
	RoBERTa-large			0.80 (0.90)	0.81 (0.91)	0.64	0.68	0.65	0.65	0.88	0.87
yahoo6	BERT-base	0.42	0.42	0.73 (0.86)	0.73 (0.87)	0.91	0.93	0.92	0.94	0.85	0.85
	BERT-large			0.58 (0.87)	0.57 (0.87)	0.82	0.81	0.80	0.81	0.88	0.84
	RoBERTa-base			0.61 (0.78)	0.79 (0.93)	0.74	0.83	0.72	0.81	0.73	0.87
	RoBERTa-large			0.62 (0.80)	0.69 (0.81)	0.79	0.79	0.79	0.81	0.78	0.78

4.6 Conclusion

We have studied the performance of embeddings obtained from four pre-trained Transformer models when used as input to a document clustering algorithm. This is a contribution to the use of Transformer representations in the unsupervised learning setting.

Our experiments show that the proposed clustering ensemble method combined with PCA-based reduction allows to make the most of Transformer-based models, achieving even better performance than that provided by the best layer (which is very difficult to identify

in an unsupervised context). We also show the advantage of using an ensemble procedure instead of end-to-end multi-view techniques. Paths for future research include continuing to improve the ensemble procedure, especially the estimation of the number of clusters and experimenting with other dimension reduction techniques. Another perspective would be to assess the impact of fine-tuning Transformer models on text clustering.

Chapter 5

On an Alternative to Fine-tuning: a Tandem Approach

Dense text representations are gaining great interest in several tasks such as Text Classification and Question Answering. However, while many challenges are faced in the unsupervised learning domain, much less is known about how suitable those various representations are when dealing with an unlabeled dataset. In this chapter, we investigate the use of such representations when performing unsupervised tasks: document clustering and visualization. Thereby, to address the document clustering objective, we propose the use of a *tandem approach* combining dimensionality-reduction techniques and clustering. We first show the benefit of relying on the subspace obtained by UMAP to perform clustering rather than using PCA-based dimension reduction. Then, through experiments performed on real datasets with static and contextual representations, we show the effectiveness of the proposed tandem approach on pre-trained representations in comparison to fine-tuning strategies proposed in the literature.

5.1 Introduction

Unsupervised learning is a ubiquitous task in data science. When dealing with an unlabeled dataset, clustering and visualization, for instance, can be very helpful to create value from textual data. In the document clustering context, however, the first problem to address is the way of representing the documents. A wide range of representations is offered to practitioners, such as Bag-Of-Words (BOW), static word embeddings like Word2vec (Mikolov et al., 2013b) and GloVe (Pennington et al., 2014) as well as contextual embeddings provided by ELMo (Peters et al., 2018b) and Transformer-based Pre-trained Language Models (T-PLM) like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019b) and DistilBERT

(Sanh et al., 2019). However, while many challenges are faced in the unsupervised learning domain, much less is known about how suitable those various representations are when dealing with an unlabeled dataset. In particular, Transformer-based embeddings are gaining more and more interest, achieving state-of-the-art results in many NLP tasks such as Question Answering, Semantic Textual Similarity (STS) and Named Entity Recognition (NER) but are much less present in the unsupervised realm of NLP. It has been shown in (Reimers and Gurevych, 2019) that the performance obtained by BERT on unsupervised STS (USTS) is poorer than those obtained by GloVe, but the study focused only on the last layer of BERT and without any post-processing, whereas it has been shown that this is far from being the best strategy to benefit fully from the T-PLM representations (Ait-Saada et al., 2021; Li et al., 2020a). Furthermore, several studies have shown that the last layers of T-PLMs tend to be task-specific (Kovaleva et al., 2019; van Aken et al., 2019) and hence perform poorly (Carlsson et al., 2021). In order to improve the semantic quality of those representations, several works propose a fine-tuning of T-PLMs on a wide range of supervised and unsupervised tasks. One of the most famous approaches is Sentence-BERT (Reimers and Gurevych, 2019) which consists in fine-tuning T-PLMs on supervised tasks and using the last layer's representation as input to unsupervised tasks. This approach is intended to be well suited to perform downstream unsupervised tasks including clustering but has not been evaluated on the latter.

In this chapter, we conduct a fully unsupervised study to determine which of those representations are the most suitable to perform document clustering. Each of them is also evaluated when post-processed using *dimensionality reduction* techniques such as PCA, t-SNE, and UMAP. This tandem approach allows to distill the information provided by pre-trained representations, thus obtaining a surprising improvement in the results along with a drastic reduction of the dimensions. This topic has not been thoroughly tackled yet in the context of text representations.

The main contributions of this study are as follows:

- We address the question of choosing the right representation to perform effective document clustering and visualization. A comparative study is performed to assess the performance of pre-trained and fine-tuned models as well as static representations.
- We evaluate different post-processing techniques based on dimension reduction, as part of a tandem clustering approach, showing that we can do better than formerly used PCA-based approaches, both in terms of clustering and 2D visualization.

5.2 Related Work

Given a text-mining task, several vector representations of the documents can be given as input, and the choice of which one to use is not straightforward, especially in an unsupervised context. Hence we propose to evaluate several text representations in terms of clustering performance. The first baselines are the classical sparse BOW and dense BOW such as Word2vec (Mikolov et al., 2013b), GloVe (Pennington et al., 2014), and fastText (Bojanowski et al., 2017), also referred to as static word embeddings, the vector of a given word being fixed regardless of its context. More recent text representations are those provided by T-PLMs like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019b) that provide token-wise representations, usually averaged together to obtain a unique representation for each document. To further improve the quality of T-PLMs, several fine-tuning strategies are proposed in the literature. Reimers and Gurevych (Reimers and Gurevych, 2019) fine-tune a siamese T-PLM on the NLI and STS tasks, thus improving the performance obtained by the last layer of BERT and RoBERTa on the USTS task. DvBERT (Cheng, 2021) also uses a siamese network, in which the sentence embeddings are augmented with word-level interaction features, requiring labeled data. On the other hand, several unsupervised approaches are proposed (Carlsson et al., 2021; Gao et al., 2021; Liu et al., 2021; Yan et al., 2021; Zhang et al., 2020b), all based on self-supervised objectives and do not require any labeled data. All the aforementioned approaches have been exclusively evaluated on the USTS task and it is not known whether they are well suited to perform document clustering.

Another way of improving the results obtained by pre-trained representations is to rely on post-processing techniques applied to the output embeddings. Those approaches mostly use PCA-based dimension reduction which has proven efficient enough to capture the semantic information present in the pre-trained representations in fewer dimensions. In the case of static word embeddings, a PCA-based approach proposed in (Raunak et al., 2019) is used to halve the dimensions without altering the performance. Regarding T-PLMs, the benefit of PCA-based dimension reduction along with a whitening step (cf. Section 5.3) has been assessed in (Huang et al., 2021; Su et al., 2021) for the USTS task and (Ait-Saada et al., 2021) for document clustering where it has shown a significant improvement in the performance.

In this study, we evaluate both the pre-trained and fine-tuned versions of the Transformer models, each one using several ways of leveraging the representations it provides. Each document's representation is obtained by an average pooling of its tokens' representations. We use the vectors provided by the last layer ("last") as performed in (Carlsson et al., 2021; Cheng, 2021; Reimers and Gurevych, 2019) as well as the combination (average) of the last two layers ("avg two last") as suggested in (Li et al., 2020a; Yan et al., 2021). We also

investigate the use of all of the layers ("avg all") as in (Ait-Saada et al., 2021). The results obtained using the last two layers are given in Figure C.1 in the Appendix. In this study, we discard the use of the [CLS] token that is still utilized in some sentence embedding approaches (Gao et al., 2021; Liu et al., 2021) but is not relevant given the models used in this work.

5.3 Tandem Approach

Given a dataset $\mathcal{D}$ of n documents, several ways of representing it are possible. In the case of a Transformer model of b layers, we end up with b different data matrices $\mathbf{X}_\ell,\ \ell \in 1, \ldots, b$ with an average pooling of the tokens (Reimers and Gurevych, 2019). We derive from those matrices one unique matrix $\mathbf{X}$ by combining a certain number of layers (as described in Algorithm 4, step 2). The tandem approach consists in combining both dimension reduction and clustering, as described in Algorithm 4. In this case, dimension reduction is seen as a post-processing step and aims at compressing $\mathbf{X}$ and improving the quality of clustering. Given a matrix $\mathbf{X}_{(n\times d)}$, we call its reduced version $\mathbf{Y}_{(n\times d')}$, $d' \ll d$. To respect the unsupervised context of the study and make a fair comparison between dimension reduction techniques, we use a constant value of $d' = 10$, no tuning of this parameter being possible in the absence of labels. Besides, dimension reduction techniques are also commonly used for visualization with $d' = 2$ where $\mathbf{Y}$ can be visualized in a 2D plane.

In previous works, post-processing of text representations has usually been performed using PCA-based dimension reduction (Ait-Saada et al., 2021; Huang et al., 2021; Raunak et al., 2019; Su et al., 2021). We define $\mathbf{X}$'s reduced representation classically derived by PCA as the projections $\mathbf{Y} = \mathbf{XQ}$, where $\mathbf{Q}$ is composed of the d' first eigenvectors of $\mathbf{X}^T\mathbf{X}$. In this study, we assess the impact of the whitening operation that has proven its effectiveness on Transformer embeddings (Ait-Saada et al., 2021; Huang et al., 2021; Su et al., 2021) but has not been compared to other dimension reduction approaches. The whitening operation consists in using $\mathbf{Y} = \mathbf{XQ}/\sqrt{\Delta}$ instead of $\mathbf{XQ}$, Δ being the first d' eigenvalues of $\mathbf{X}^T\mathbf{X}$. Another way of reducing the dimension used in this work is UMAP (McInnes et al., 2018), which is a nonlinear manifold technique of dimension reduction that starts with the construction of a graph that approximates the structure of the data points in the original space, followed by a projection into a lower dimension space. UMAP has proven effective in improving several tasks including clustering (Allaoui et al., 2020; McConville et al., 2021). We preferred using UMAP instead of t-SNE (van der Maaten and Hinton, 2008) due to the computation restrictions of t-SNE that make it unsuitable for post-processing with $d' > 3$. Another notable difference is that UMAP yields a better balance between local

Algorithm 4: Tandem Approach on Transformers

input : $\mathcal{D}$ a dataset of n documents; $\mathcal{M}$ a Transformer model, $\mathcal{L}$ a set of layer indices, $\mathcal{C}$ a clustering algorithm; k the number of clusters; d' the number of dimensions, $\mathcal{R}$ a dimension reduction function, $\mathcal{Q}$ a set of t label initializations of size n

output : A clustering partition $\mathbf{p}$

Compute the $\mathbf{X}^{\ell}$ matrix representations using $\mathcal{M}$, $\ell \in \mathcal{L}$;
$\mathbf{X} \leftarrow \textit{average}(\mathbf{X}^{\ell})$, $\ell \in \mathcal{L}$;
$\mathbf{Y} \leftarrow \mathcal{R}(\mathbf{X}, d')$;
$i \leftarrow 1$;
foreach $\mathbf{q} \in \mathcal{Q}$ **do**
    $\mathbf{p}^{(i)} \leftarrow \mathcal{C}(Y, k, \mathbf{q})$;
    $\mathbf{c}^{(i)} \leftarrow \textit{criterion}\,(\mathcal{C}, \mathbf{p}^{(i)})$;
    $i \leftarrow i + 1$;
end
$i^* \leftarrow \underset{i \in 1,\ldots,t}{argmax}(c^{(i)})$
$\mathbf{p} \leftarrow \mathbf{p}^{(i^*)}$;
return $\mathbf{p}$;

and global structures. Especially, UMAP efficiently preserves the global structure thanks to its topological approximation based on Reimannian geometry assumptions.

5.4 Experimental Study

In this study, six different models are used: BERT and RoBERTa, SBERT, and SRoBERTa (Reimers and Gurevych, 2019), SBERT-CT and SRoBERTa-CT (fine-tuned using the unsupervised Contrastive Tension objective (Carlsson et al., 2021)). We focus on "large" models with $b = 24$ and $d = 1024$. In addition, those representations are compared to several baselines namely BOW weighted with TFIDF of size $n \times v$, v being the size of the vocabulary as well as Word2vec, GloVe, and fastText, all of which provide $n \times d$ matrices with $d = 300$. As regards data, we use the exact same datasets used in Chapter 4 (cf. Table 4.1).

All the clustering experiments are carried out using k-means (MacQueen et al., 1967), except for the BOW representation for which Spherical k-means (Buchta et al., 2012) is preferred, leading to significantly better results than k-means, due to the directional nature of the obtained data matrix. We perform clustering over $t = 30$ different initializations and keep the one providing the highest value of within-cluster inertia, as described in Algorithm 4, where $\mathcal{C}$ is set to k-means. In order to assess the quality of the clustering results, we rely

Table 5.1: Clustering scores (NMI in %) obtained using different strategies applied to BERT-large text representations.

	Dataset	BOW	Static		Pre-trained	Fine-tuning		Tandem		
			fastText	GloVe	BERT	SBERT	SBERT-CT	PCA	PCA_w	UMAP
All layers	classic3	0.952	0.795	0.887	0.9	0.895	0.887	0.899	0.948	**0.964**
	classic4	0.689	0.218	0.547	0.68	0.462	0.643	0.681	0.712	**0.744**
	BBC	0.81	0.439	0.738	0.788	0.693	0.748	0.763	0.797	**0.867**
	AG-news	0.487	0.03	0.529	0.484	0.355	0.516	0.439	0.47	**0.556**
	DBPedia	0.714	0.452	0.727	0.698	0.532	0.634	0.619	0.613	**0.74**
Last layer	classic3	0.952	0.795	0.887	0.932	0.873	0.912	0.931	0.949	**0.971**
	classic4	0.689	0.218	0.547	0.202	0.607	0.658	0.202	0.693	**0.728**
	BBC	0.81	0.439	0.738	0.768	0.799	**0.815**	0.756	0.757	0.779
	AG-news	0.487	0.03	**0.529**	0.002	0.197	0.416	0.002	0.208	0.196
	DBPedia	0.714	0.452	**0.727**	0.562	0.413	0.567	0.437	0.437	0.553

Table 5.2: Clustering scores (NMI in %) obtained using different strategies applied to RoBERTa-large text representations.

	Dataset	BOW	Static		Pre-trained	Fine-tuning		Tandem		
			fastText	GloVe	RoBERTa	SRoB.	SRoB-CT	PCA	PCA_w	UMAP
All layers	classic3	0.952	0.795	0.887	0.864	0.861	0.896	0.861	0.928	**0.956**
	classic4	0.689	0.218	0.547	0.51	0.645	0.687	0.505	0.695	**0.737**
	BBC	0.81	0.439	0.738	0.741	0.701	0.845	0.736	0.838	**0.892**
	AG-news	0.487	0.03	0.529	0.513	0.548	**0.575**	0.497	0.567	0.533
	DBPedia	0.714	0.452	**0.727**	0.687	0.696	0.711	0.605	0.629	0.687
Last Layer	classic3	0.952	0.795	0.887	0.915	0.848	0.904	0.908	0.959	**0.983**
	classic4	0.689	0.218	0.547	0.719	0.634	0.673	0.711	0.721	**0.751**
	BBC	0.81	0.439	0.738	0.874	0.573	0.835	0.864	0.764	**0.907**
	AG-news	0.487	0.03	**0.529**	0.501	0.336	0.408	0.466	0.518	0.509
	DBPedia	0.714	0.452	**0.727**	0.715	0.623	0.704	0.574	0.598	0.676

on Normalized Mutual Information (NMI) (Strehl and Ghosh, 2002) which is a standard external measure that is less sensitive to the class imbalance in comparison to accuracy.

5.4.1 Document Clustering

Tables 5.1 and 5.2 show the clustering performance (NMI) obtained using different strategies. The following can be observed:

- The BOW model shows its limits against Word2vec and GloVe when UMAP is used. Word2vec and GloVe even show competitive results compared to Transformers except in the case of DBpedia and AG-news where the clusters are ill-separated.
- The UMAP-based approach with the RoBERTa model appears to be the wisest choice for clustering with all strategies (last layer and all layers). It shows better performance compared to BERT, with and without fine-tuning.

- Using only a few components of PCA (linear method) does not alter considerably the results and even improves the clustering quality when the whitening is used (PCA_w), though leading to a lesser improvement in comparison to UMAP (nonlinear method).
- Neither supervised (Reimers and Gurevych, 2019) nor unsupervised (Carlsson et al., 2021) fine-tuning approaches bring any significant improvement to text clustering and are even outperformed by pre-trained models. For example, when we look at the last layer (as used in (Reimers and Gurevych, 2019)), SRoBERTa performs significantly poorer than RoBERTa almost in all experiments. This might be due to the fact that we are dealing with long documents whereas SBERT and SRoBERTa are trained on NLI and STS, both involving short sentences.

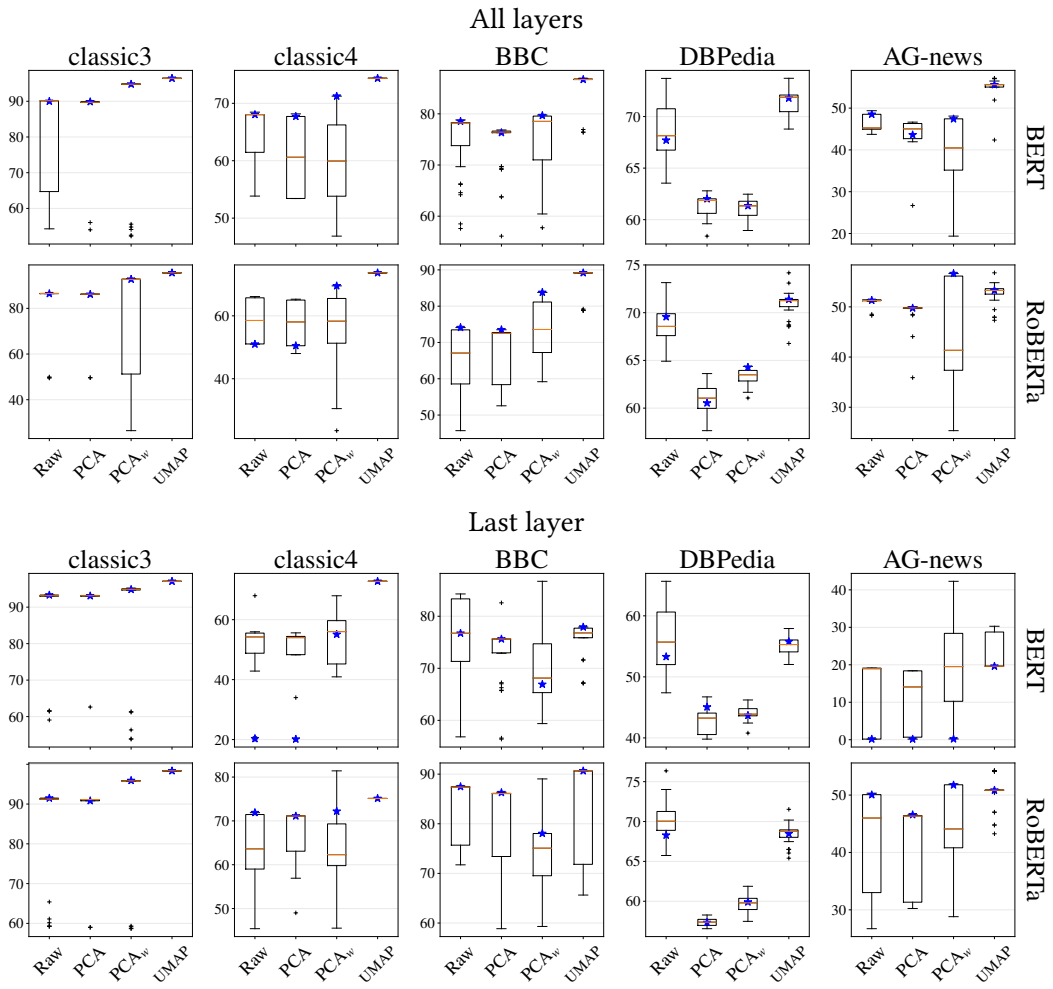

Figure 5.1: Distribution of the NMI (in %) over 30 initializations. The blue star symbol corresponds to the score of the solution selected using k-means' criterion (values given in Tables 5.1 and 5.2) and the orange line to the median.

Fig. 5.1 shows the distribution of NMI obtained with different approaches. We first observe the advantage of using all of the layers instead of using only the last one, in terms of clustering quality and also in terms of robustness. We indeed notice that the NMI score is much less dependent on the initialization when using all of the layers. Further, we can see from Fig. 5.1 how robust the representations provided by UMAP are, especially in comparison to PCA_w which presents a much higher variance along with a significantly lower median. This shows the reliability and robustness of using UMAP as part of the tandem approach in comparison to the PCA-based approaches and the use of the original representations.

5.4.2 Data Visualisation

Fig. 5.2 shows 2D projections obtained using different dimension reduction techniques with $d' = 2$. The given NMI score is computed using the real labels and the agreement "Agr" is an unsupervised measure proposed in (France and Akkucuk, 2021) which quantifies the agreement between the original and the latent space. It is computed as:

$$Agr_k = \frac{1}{kn} \sum_{i=1}^{n} \left[a_{ik} - \frac{k}{n-1} \right] \tag{5.1}$$

where k is the neighborhood size parameter that we vary between 1 and 100 while a_{ik} represents the shared elements by the first k columns of the ranking matrices $\mathbf{N}_{\mathbf{X}(n\times n)}$ and $\mathbf{N}_{\mathbf{Y}(n\times n)}$ containing in each row i the indices of samples from the closest to the farthest w.r.t. a certain metric (euclidean in our case), using $\mathbf{X}$ and $\mathbf{Y}$ resp.

Due to the difficulty to tune dimension reduction's hyperparameters of t-SNE and UMAP, we set the perplexity and the number of nearest neighbors to 15 and retain the euclidean distance. Overall, we observe the difference in terms of cluster separability. Thereby, UMAP is capable of better separating the clusters followed by t-SNE then PCA which, hardly surprisingly, shows very poor separability of clusters. This is corroborated by the NMI score, which is always higher for UMAP. To go further, the agreement score is much lower for PCA, however, it is higher for t-SNE in comparison to UMAP. This means that t-SNE respects better the original local structure of data. However, t-SNE does not allow a better separability of clusters, which makes UMAP a good trade-off between data embedding and separability between clusters. This also suggests that the distortion brought by UMAP is more beneficial to clustering and shows how UMAP better respects the global data structure in the latent space.

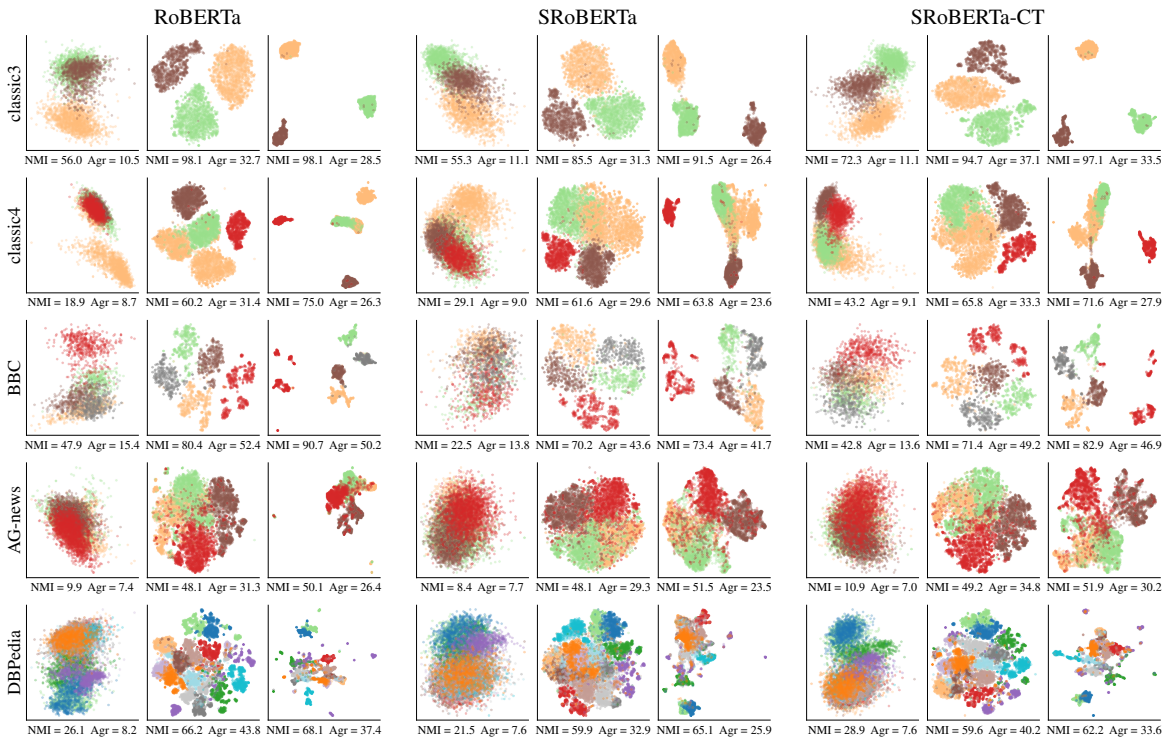

Figure 5.2: 2D projections obtained by PCA_w, t-SNE and UMAP respectively. The NMI score (%) corresponds to the clustering performance applied to the reduced version $\mathbf{Y}_{(n\times d')}$ with $d' = 2$. The Agr score (%) is the agreement measure between the original and reduced space. Data points are colored according to the real classes. This graphics confronts three versions of the same model: RoBERTa-large.

5.4.3 What about Simultaneous Approaches?

The tandem approach used so far is usually opposed to simultaneous (or end-to-end) methods that optimize dimension reduction clustering in a concomitant way. We have shown via the empirical study conducted in the previous chapter that simultaneous techniques are not necessarily better than multi-stage approaches, especially on the dense representations delivered by pre-trained language models. In this section, we compare two-way end-to-end techniques with tandem approaches using the same size for the embedding space.

The first approach we use is Reduced k-means (RKM, De Soete and Carroll 1994) which combines a linear dimension reduction with the k-means criterion (cf. Section 1.3.4). We also evaluate two previously mentioned approaches based on deep neural networks namely DCN (Yang et al., 2017) and DKM (Fard et al., 2020) that jointly optimize the reconstruction loss of an autoencoder and a clustering loss based on k-means. For both models, we use the same architecture as in (Fard et al., 2020) and vary the values of the hyperparameter λ between 10^{-4} and 10^{3} but instead of using the best value for each dataset, we choose

Table 5.3: NMI scores obtained by simultaneous and tandem approaches. The **bold** numbers correspond to the best score in each row and the underlined numbers are for the second-best performance score.

Dataset	Model	Simultaneous			Tandem		
		RKM	DCN	DKM	PCA	PCA_w	UMAP
classic3	BERT	0.91	0.93	0.95	0.9	0.95	**0.96**
	RoBERTa	0.87	0.86	0.89	0.86	0.93	**0.96**
classic4	BERT	0.68	0.66	0.7	0.68	0.71	**0.74**
	RoBERTa	0.66	0.62	0.69	0.5	0.7	**0.74**
BBC	BERT	0.79	0.78	0.83	0.76	0.8	**0.87**
	RoBERTa	0.74	0.6	0.65	0.74	0.84	**0.89**
DBPedia	BERT	0.66	0.64	0.67	0.62	0.61	**0.74**
	RoBERTa	0.61	0.69	**0.73**	0.6	0.63	0.69
AG-news	BERT	0.45	0.54	**0.57**	0.44	0.47	0.55
	RoBERTa	0.52	0.57	**0.58**	0.5	0.57	0.53

the one that yields the best overall results $\lambda = 1$ for DKM and $\lambda = 0.01$ for DCN. Note that this gives an advantage to DKM and DCN since RKM and PCA do not require any hyperparameter tuning and that we use the recommended hyperparameters of UMAP instead of performing any tuning. We use the TensorFlow implementation of DCN and DKM provided by Fard et al.[1] and our implementation of RKM. Table 5.3 contains the NMI score obtained with simultaneous approaches. We first observe that combining PCA and k-means simultaneously (RKM) achieves better results than the tandem approach with PCA. However, PCA_w and UMAP outperform RKM by far. Particularly, UMAP achieves the best performance scores and beats all the simultaneous techniques except on AG-news and DBPedia using RoBERTa. Among the simultaneous approaches, DKM yields the best overall results, particularly on AG-news.

5.5 The Real Impact of Anisotropy

In the last few years, several studies have been devoted to dissecting dense text representations in order to understand their effectiveness and further improve their quality. Particularly, the anisotropy of such representations has been observed, which means that the directions of the word vectors are not evenly distributed across the space but rather concentrated in a narrow cone. This has led to several attempts to counteract this phe-

[1] https://github.com/MaziarMF/deep-k-means

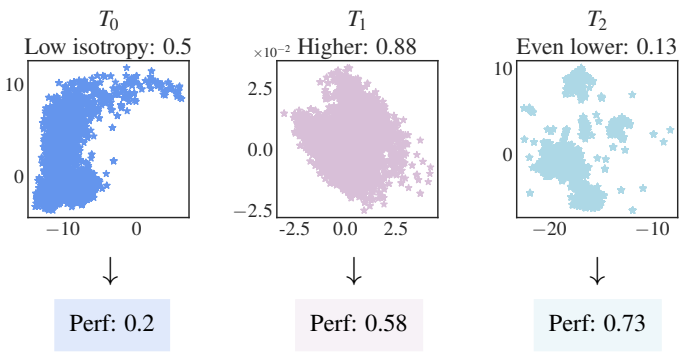

Figure 5.3: Isotropy versus performance with different transformations (T_0 = no transformation).

nomenon both on static and contextualized text representations. However, despite this effort, there is no established relationship between anisotropy and performance. In this section, we aim to bridge this gap by investigating the impact of the transformations used in the previous section on both the isotropy and the performance in order to assess the true impact of anisotropy. To this end, we rely on the clustering task as a means of evaluating the ability of text representations to produce meaningful groups. Thereby, we empirically show a limited impact of anisotropy on the expressiveness of sentence representations both in terms of directions and L_2 closeness.

5.5.1 Background

It is now well established that language models in general (Gao et al., 2019) and Transformer word embedding models in particular (Ethayarajh, 2019; Wang et al., 2020) produce an anisotropic embedding space. This concretely means that the directions of trained dense word representations do not occupy uniformly the embedding space, which is suspected to limit their expressiveness and thus their expected performance on downstream tasks. The main question addressed in this section is how harmful this anisotropy really is regarding the quality of text representations.

Several approaches have been proposed to increase the isotropy of dense representations, based on different strategies. In the context of static word embeddings like GloVe and word2vec, both Raunak et al. (2019) and Mu and Viswanath (2018) proposed a post-processing method that consists in removing the first principal components before reconstructing the word vectors as opposed to the traditional approach of removing the weakest components. This approach improves the quality of word vectors on several downstream tasks while reducing their anisotropy (Mu and Viswanath, 2018).

As to contextualized representations provided by Transformer models, several approaches have been proposed in order to alleviate the anisotropy problem. For instance, based on the idea that anisotropic representations tend to have high expected pairwise cosine similarity, Wang et al. (2020) proposed to apply a cosine similarity regularization term to the embedding matrix. In the same vein, Gao et al. (2019) proposed a method named "spectrum control" that allows for increasing the isotropy of Transformer representations and improving the performance of the machine translation task. To this purpose, they propose regularization terms that hamper the singular value decay of the embedding matrix. However, despite the success of these *optimization* tricks in lowering the anisotropy of Transformer representations, Ding et al. (2022b) have recently shown that they do not bring any improvement, relying on several tasks like summarization and sentence similarity. They even observed a certain deterioration of the performance brought by anisotropy mitigation techniques.

In contrast, Rajaee and Pilehvar (2021, 2022) showed that *post-processing* methods made for increasing isotropy are also responsible for a performance increase in the sentence similarity task in both monolingual and cross-lingual settings. Similarly, the whitening operation, which consists in using the principal components normalized by their inertia, has shown an increase in isotropy as well as a better performance in sentence similarity (Su et al., 2021) and clustering (Ait-Saada et al., 2021). However, there is no evidence that the decrease of anisotropy brought by such transformations is directly responsible for the gain of performance, as shown in Figure 5.3, which gives a first idea of the question addressed in this section.

Indeed, despite the great energy devoted to studying and mitigating the anisotropy of dense text representations, there is no clear connection between isotropy and performance, which seems to depend, inter alia, on the sought task. In order to contribute to settling this question, we consider using a task that has never been used for this purpose: document clustering. The rationale behind this choice is to evaluate, under different degrees of isotropy, the capability of text representations to facilitate the clear separation and identification of meaningful groups of documents through clustering algorithms. The main contributions of this study are:

- We extend the isotropy study of word embeddings to document representations.
- We investigate the correlation between different isotropy measures.
- We assess the connection between isotropy and quality of representation.

5.5.2 Isotropy and Performance: Experimental Study

In this study, we aim to determine to what extent the anisotropy actually affect the quality of the representations and their ability to discriminate data samples through separable clusters. To this end, we use both measures to evaluate the anisotropy of the original embedding space before and after post-processing. We also compare the changes in isotropy with the corresponding cluster performance in order to establish a potential relationship between the two concepts.

Relying on several isotropy measures allows us to consolidate confidence in our conclusions and, at the same time, verify if the measures agree with each other. In the same spirit, using different clustering methods and performance measures insures more rigorous assertions.

Isotropy measures

Let $\mathcal{X} = \{\mathbf{x}_i\}$ be a set of v vector representations, characterizing either v words or v documents. In (Mu and Viswanath, 2018), the isotropy is assessed using the partition function ψ as follows:

$$\frac{\min_{\|\mathbf{c}\|=1} \psi(\mathbf{c})}{\max_{\|\mathbf{c}\|=1} \psi(\mathbf{c})}; \quad \text{where } \psi(\mathbf{c}) = \sum_{i=1}^{v} e^{\langle \mathbf{x}_i, \mathbf{c} \rangle}$$

This approach is inspired by the theoretical findings issued by Arora et al. (2016) who proved that, for isotropic representations $\mathcal{X}$, the partition function ψ can be approximated by a constant for any unit vector $\mathbf{c}$, thus leading to a min/max ratio score of 1. As there is no analytic solution $\mathbf{c}$ that maximizes or minimizes $\psi(\mathbf{c})$, Mu and Viswanath proposed to use the eigenvectors of the covariance matrix as the set of unit vectors, which leads to:

$$\mathcal{I}_{pf}(\mathcal{X}) = \frac{\min_{\mathbf{w}_j} \psi(\mathbf{w}_j)}{\max_{\mathbf{w}_j} \psi(\mathbf{w}_j)} \tag{5.2}$$

where *pf* stands for the *partition function*, $\mathbf{w}_j$ is the jth eigenvector of $\mathbf{X}^\top\mathbf{X}$ ($\mathbf{X}$ being the representation matrix). In our experiments, $\mathcal{X}$ contains representations of either words or sentences/documents. In addition to this measure, (Wang et al., 2020) quantify the anisotropy by the standard deviation of the partition function normalized by the mean:

$$\mathcal{A}(\mathcal{X}) = \sqrt{\frac{\sum_{j=1}^{d} (\psi(\mathbf{w}_j) - \bar{\psi}))^2}{d\,\bar{\psi}^2}}$$

where $\bar{\psi}$ is the average value of the partition function. Perfectly isotropic representations lead to $\mathcal{A}(\mathcal{X}) = 0$ and greater values denote a higher anisotropy. For our purpose, we derive the isotropy score as the square root of the precision score $\tau = 1/\sigma$, which leads to:

$$\mathcal{I}_{pf_2}(\mathcal{X}) = \frac{1}{\sqrt{\sigma}} = \frac{1}{\mathcal{A}(\mathcal{X})}$$

σ being the variance normalized by $d\bar{\psi}^2$.

On the other hand, the study of (an)isotropy provided in (Ethayarajh, 2019) has been applied to word representations and the empirical results have been obtained using a high number of words picked randomly. The authors relied on the assumption that the expected similarity of two words uniformly randomly sampled from an isotropic embedding space is zero and that high similarities thus denote an anisotropic embedding space. They hence use the average cosine similarity between randomly sampled words in order to assess the anisotropy level of word representations. The isotropy is defined as:

$$\mathcal{I}_{cos} := \mathbb{E}_{i \neq i'}(1 - \cos(\mathbf{x}_i, \mathbf{x}_{i'})) \tag{5.3}$$

where the score is computed over m random pairs $(\mathbf{x}_i, \mathbf{x}'_i)$ of vector representations.

Quality measures

In order to assess the quality of text representations $\mathcal{X}$, we rely on the clustering task. To this end, we estimate the ability of a clustering algorithm to accurately distinguish groups of documents in a corpus, represented by $\mathcal{X}$. This is achieved using two well-known measures: Normalized Mutual Information (NMI, Strehl and Ghosh 2002), the Adjusted Rand Index (ARI, Hubert and Arabie 1985; Steinley 2004). Both are often used to compare two clustering partitions, even when the number of clusters is different (cf. Section 1.2.7).

Euclidean vs. cosine

As a recall, anisotropic vector directions occupy a narrow cone in the geometrical space. Given this definition, we can expect directional techniques based on the angles between vectors to be particularly sensitive to the alleged lack of expressiveness induced by anisotropy. With this in mind, we use Spherical k-means (Dhillon and Modha, 2001), a variant of k-means made for directional data and based on the cosine distance instead of the L_2 metric. For both clustering algorithms, we use 10 different initialization and keep the partition that yields the best within-cluster inertia.

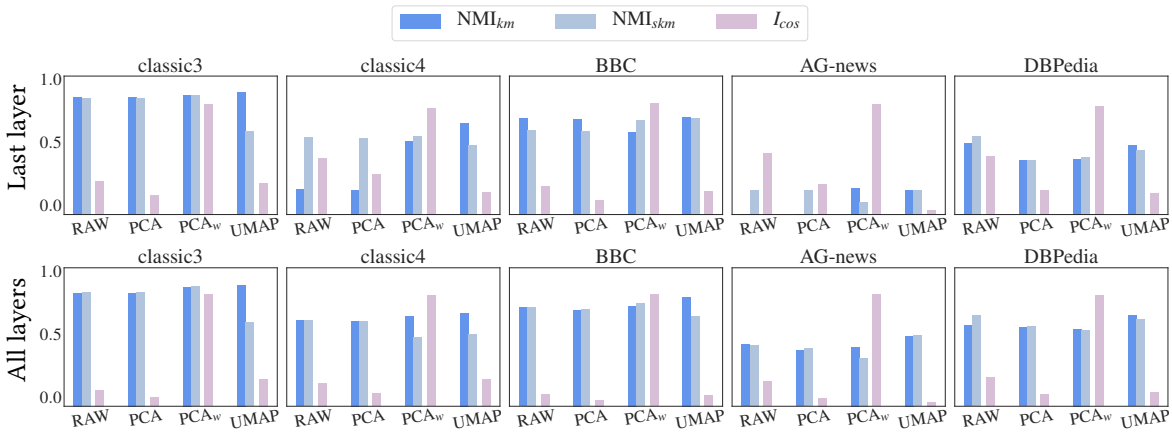

Figure 5.4: Isotropy against clustering performance. The first row is obtained using the last layer of BERT while the second row uses all the layers averaged together. NMI$_{km}$ and NMI$_{skm}$ correspond to the NMI score obtained by k-means and spherical k-means respectively. $\mathcal{I}_{cos}$ represents the cosine isotropy score computed using Equation 5.3.

Table 5.4: Pearson correlation coefficient values between several isotropy and performance measures. The corresponding p-values are given in Table C.1 in the Appendix. "Dataset" means that the isotropy has been computed within the same dataset on which NMI and ARI are computed. "External" means that the isotropy has been evaluated using an external dataset either at the "word" or "sentence" level.

		NMI		ARI		Dataset			External (word)			External (sentence)		
		km	*skm*	*km*	*skm*	*cos*	*pf*	*pf*$_2$	*cos*	*pf*	*pf*$_2$	*cos*	*pf*	*pf*$_2$
NMI	*km*	1.0	0.83	0.94	0.74	-0.05	0.02	-0.05	-0.14	0.02	0.02	-0.07	0.01	0.01
	skm	0.83	1.0	0.74	0.92	-0.09	-0.05	-0.15	-0.05	-0.05	-0.05	-0.08	-0.06	-0.05
ARI	*km*	0.94	0.74	1.0	0.78	-0.07	0.01	-0.07	-0.12	0.01	0.01	-0.07	-0.0	0.01
	skm	0.74	0.92	0.78	1.0	-0.01	0.04	-0.07	0.07	0.04	0.04	0.02	0.04	0.04
Dataset	*cos*	-0.05	-0.09	-0.07	-0.01	1.0	0.96	0.9	0.84	0.95	0.95	0.98	0.96	0.95
	pf	0.02	-0.05	0.01	0.04	0.96	1.0	0.95	0.76	1.0	1.0	0.94	1.0	1.0
	pf$_2$	-0.05	-0.15	-0.07	-0.07	0.9	0.95	1.0	0.73	0.95	0.95	0.89	0.95	0.95
Ext-w	*cos*	-0.14	-0.05	-0.12	0.07	0.84	0.76	0.73	1.0	0.76	0.76	0.89	0.78	0.75
	pf	0.02	-0.05	0.01	0.04	0.95	1.0	0.95	0.76	1.0	1.0	0.94	1.0	1.0
	pf$_2$	0.02	-0.05	0.01	0.04	0.95	1.0	0.95	0.76	1.0	1.0	0.94	1.0	1.0
Ext-s	*cos*	-0.07	-0.08	-0.07	0.02	0.98	0.94	0.89	0.89	0.94	0.94	1.0	0.96	0.93
	pf	0.01	-0.06	-0.0	0.04	0.96	1.0	0.95	0.78	1.0	1.0	0.96	1.0	1.0
	pf$_2$	0.01	-0.05	0.01	0.04	0.95	1.0	0.95	0.75	1.0	1.0	0.93	1.0	1.0

5.5.3 Discussion

Figure 5.4 confronts one quality measure (NMI) and one isotropy measure ($\mathcal{I}_{cos}$) using different post-processing techniques. We first observe that PCA$_w$ produces, by far, the most isotropic representations while increasing the performance of the raw vectors. Indeed, an appealing explanation of the success of the whitening operation is that it considerably alleviates the anisotropy of the embedding space (Su et al., 2021). Applying that reasoning, PCA and UMAP should deteriorate the performance since they both exacerbate the anisotropy

(in all cases for PCA and in most cases for UMAP). Nonetheless, the performance of PCA is comparable to that of the raw embeddings and UMAP achieves even better performance than PCA_w even though it significantly reduces the isotropy. Overall, averaging the whole set of layer representations achieves better results, even though it clearly decreases the isotropy, compared to using the last layer as traditionally performed. Also, it is worth noting that even when the *directions* of the vectors are used (skm), the decrease in isotropy has a negligible impact on the performance. All these observations suggest that, although the anisotropy reduces the spectrum of directions taken by sentence vectors, it does not necessarily alter their expressiveness.

In order to confirm this supposition, we directly compare isotropy and quality measures in a wide range of situations. To this end, we compute the correlation between several isotropy measures and performance scores on 2 models (BERT and RoBERTa) with 2 different strategies ("all layers" and "last"), using 5 datasets and 4 transformations, leading to a total of 80 occurrences for each measure. For external evaluation of isotropy, we make use of the dataset used by Rajaee and Pilehvar (2022) which contains sentences extracted from Wikipedia. We use this dataset to evaluate the isotropy measures like $\mathcal{I}_{cos}$, computed between $m = 5\,000$ pairs of words and sentences. The $10\,000$ resulting representations are also used to compute $\mathcal{I}_{pf}$ and $\mathcal{I}_{pf_2}$.

To assess the linear correlation between two measures, we use the Pearson correlation coefficient ρ (Pearson, 1896) and test its significance. The ρ coefficient between two random variables X and Y indicates how much one of the variables increases with the growth of the other. It is computed as:

$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sqrt{\sigma_X \sigma_Y}}$$

where X et Y are two random variables of variance σ_X et σ_Y respectively and $\text{cov}(X, Y)$ is the covariance between X and Y. In order to test the significance of ρ we rely on the p-value which is a probability that denotes how likely it is that the observed variables have occurred under the null hypothesis which is that the two variables are perfectly correlated ($\rho_{X,Y} = 0$). Thus, high $\rho_{X,Y}$ values indicate a stronger linear relationship and the closer the p-value gets to zero, the more we consider significant the correlation between X and Y. The values of correlation are given in Table 5.4 and the associated p-values in Table C.1 in the Appendix.

From Table 5.4, we first observe a high correlation (associated with a near-zero p-value in Table C.1) between measures within the same family (e.g. $\mathcal{I}_{cos}$ and $\mathcal{I}_{pf}$). This indicates that the selected measures agree with each other which denotes a certain coherence. However, when looking at the correlation between the two families of measures, it is clear that there

is no significant relationship between isotropy and quality measures, since all the values of the correlation coefficient are close to zero, which is corroborated by relatively high p-values, denoting a non-significant correlation. Note that the same observations can be made using the Spearman correlations of ranks (Spearman, 1987).

5.6 Conclusion

As T-PLM representations showed poor performance on downstream tasks, several approaches have been proposed in order to address this issue. These approaches are based either on fine-tuning the pre-trained models or post-processing the output representations, or combining both operations. In this chapter, we assess the impact of such approaches on two tasks – text clustering and 2D visualization. Thereby, we show that fine-tuning, even though beneficial to the USTS task, does not bring any significant improvement in either of the two tasks. Post-processing, however, in addition to being simpler, shows much more impressive improvements. More specifically, we highlight the potential of UMAP. Even with a low-dimensional subspace, UMAP shows a surprising improvement in the clustering performance.

On another note, it has been known to happen that transformations that tend to decrease the anisotropy of text representations also improve the performance of downstream tasks. In stark contrast, we observe in the present study that transformations that exacerbate the anisotropy phenomenon may also improve the results, which calls into question the importance of isotropy in text representation. To draw this important conclusion, we relied on the clustering task and several empirical measures to assess the relationship between isotropy and quality of representations, using several datasets. Most importantly, we show that even a directional approach for clustering, which should be primarily affected by anisotropy, does not undergo any performance loss resulting from low-isotropy representations. In addition, we show the advantage of using UMAP as a post-processing step, which provides good-quality representations using only a handful of dimensions, despite a high resulting anisotropy.

This opens an interesting path of research to which we intend to dedicate our future work, especially on how to benefit from the non-linearity and strong mathematical foundations of UMAP in a parametric fashion using neural networks.

Chapter 6

Unsupervised Anomaly Detection in Multi-Topic Short Text Corpora

Unsupervised anomaly detection seeks to identify deviant data samples in a dataset without using labels and constitutes a challenging task, particularly when the majority class is heterogeneous. This chapter addresses this topic for textual data and aims to determine whether a text sample is an outlier within a potentially multi-topic corpus. To this end, it is crucial to grasp the semantic aspects of words, particularly when dealing with short texts, since it is difficult to syntactically discriminate data samples based only on a few words. Thereby we make use of word embeddings to represent each sample by a dense vector, efficiently capturing the underlying semantics. Then, we rely on the Mixture Model approach to detect which samples deviate the most from the underlying distributions of the corpus. Experiments carried out on real datasets show the effectiveness of the proposed approach in comparison to state-of-the-art techniques both in terms of performance and time efficiency, especially when more than one topic is present in the corpus.

6.1 Introduction

Anomaly Detection (AD) is a task that can address various objectives such as mining frauds (Deng and Mei, 2009), diseases (Han et al., 2021) and intrusions (Pu et al., 2021). AD takes several forms: supervised, unsupervised or semi-supervised. Unsupervised AD implies that no prior information about the dataset is provided. In this case, the solution usually consists of identifying samples that deviate in a certain way from the others among the same dataset; anomalies being, by definition, rare phenomena. Particularly, anomalies in a textual dataset can be defined as samples having an atypical vocabulary (lexical anomaly) or a deviating global meaning (semantic anomaly). Identifying abnormalities in textual data can

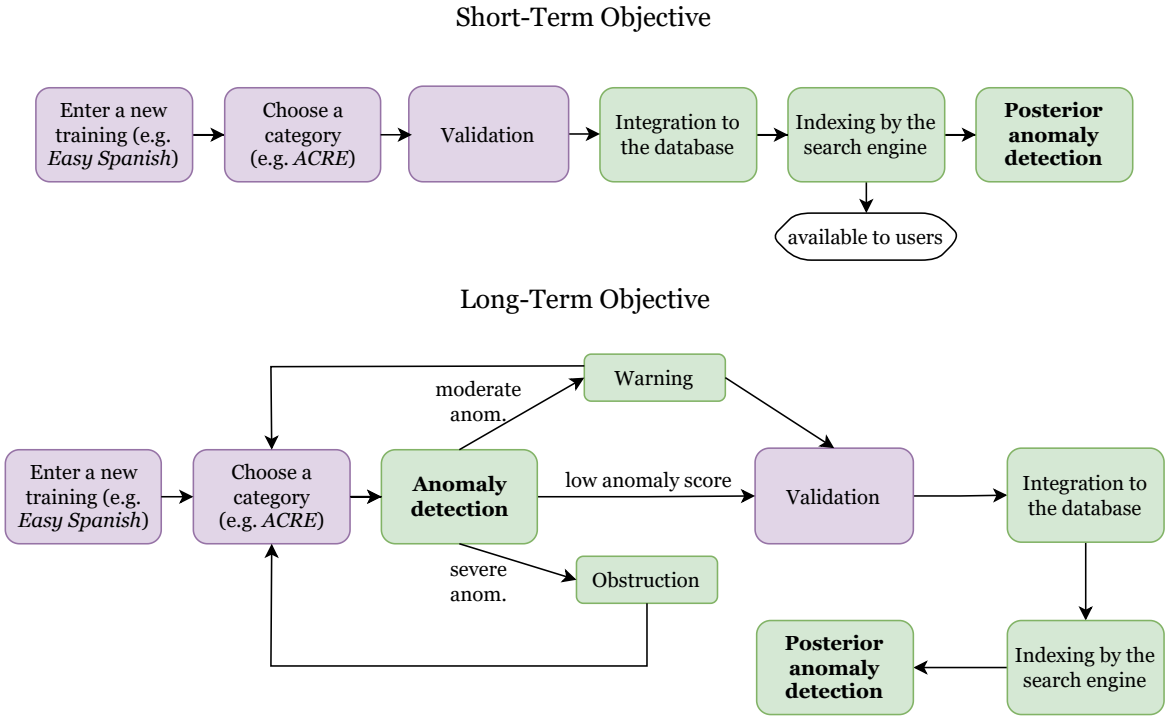

Figure 6.1: Problem statement of the industrial application of anomaly detection. Green boxes correspond to internal actions while purple boxes are for actions made by the training providers.

be very useful in many industrial use-cases. A good example is the detection of non-eligible and/or fraudulent course contents in the public French plateform MonCompteFormation[1] where millions of course sessions are available with no possibility of controlling training organizations in a supervised fashion (using labeled data). Hence, to assess the effectiveness of our approach, we rely on an external labeled dataset that closely relates to course contents and that is dedicated to course certifications. The dataset is described in Section 6.4.1. In addition to the difficulty of mining anomalies in short-text corpora of varying sizes, we also have an important computational cost constraint that is also addressed by the proposed solution.

Over the last years, a great deal of effort has been devoted to combat fraud in MCF, using several types of data such as the temporal evolution of the turnover of training providers, the interactions between the users and the providers, etc. One important kind of fraud is when training providers propose fake or ineligible training contents. We include in this context trainings that are put in the wrong category whether on purpose or not. This present work is part of a project with short-term and long-term purposes, as described in Figure 6.1. In

[1] https://www.moncompteformation.gouv.fr/ (MCF) developed by « Caisse des Dépôts et Consignations » (CDC)

the short term, our objective is to identify in the database atypical training contents that can be either fake, ineligible or wrongly categorized, so that the corresponding trainings may be corrected or removed. In the longer term, the goal is to work towards a real-time anomaly detection system that will be included in the phase during which a training provider fills the information about the training. This will help prevent the occurrence of anomalies in the database, which will in turn prevent the users of the plateform from encountering trainings that should not be there. This long-term objective motivates us to pay extra attention to the computational efficiency, as the anomaly detection system must provide real-time outputs within a short timeframe.

Table 6.1: Example illustrating the importance of semantic representations when dealing with a small corpus of short texts.

Les lignes de commande Linux pour débutants	inlier
Formation: Introduction au Shell Bash	inlier
Administration système Unix pour les nuls	inlier
Apprendre à utiliser le terminal Ubuntu/Debian	inlier
Formation en Espagnol pour débutants	outlier

Capturing the semantics of a given text is usually performed using Word Embeddings, which consist in representing a word or a piece of text by a fixed-size vector, supposed to detain its meaning. Several word embedding techniques are available such as word2vec (Mikolov et al., 2013b), GloVe (Pennington et al., 2014), fastText (Bojanowski et al., 2017), BERT (Devlin et al., 2019), DistilBERT (Sanh et al., 2019), etc. Each of the mentioned works provides ready-to-use models that are pre-trained on very large corpora and intended to be general for a given language and suitable for several NLP downstream tasks. Indeed, relying on such pre-trained models has proved efficient in several tasks (Das et al., 2017; Kim, 2014) and is particularly useful when dealing with small corpora (Buechel et al., 2018). If we consider the small corpus given in Table 6.1, we can observe that the inlier samples (training titles about Linux shell programming) do not have any words in common. Thus, based only on the syntactic information of the samples, it would be impossible to isolate the outlier (about learning Spanish), even though it is the only one that does not have anything to do with shell programming. This can easily occur when dealing with short-text corpora, especially when the number of samples is not sufficient to learn the different syntactic variants of a word or a concept.

Depending on the data type and the assumptions that can be made, the definition of an outlier may differ, and the choice of the model is crucial, especially in an unsupervised context (Aggarwal, 2017). In this chapter, we propose a probabilistic anomaly detection

methodology based on Mixture Models, that effectively identifies the most deviating samples in short-text datasets, even in the case where several topics are present in the inlier class. We also show the effectiveness of using the knowledge learned by word embedding models in capturing the underlying semantics of short texts and efficiently identifying outliers. The main contributions of this study are:

- We address the challenge of mining anomalies in short texts written in french
- We tackle the classical one-class inlier scheme and also a more challenging multi-class inlier setting.
- We propose an effective anomaly detection approach that outperforms previously proposed anomaly detection techniques in both scenarios.

6.2 Related Work

Anomaly detection is an active research area, and a large number of approaches are proposed in several application domains. Specifically, our work relates to unsupervised anomaly detection for text and clustering-based anomaly detection. Unsupervised anomaly detection is gaining more and more interest in research due to the constant growth of data volumes while labeling data samples is not getting any cheaper. One of the most important family of methods contains reconstruction-based approaches that assume that a well-generalizing model would struggle at compressing rare anomalous samples. This kind of approaches include linear models such as PCA (Jablonski et al., 2015) and deep autoencoder models based on convolutional networks (Oza and Patel, 2018), recurrent networks (Hsieh et al., 2019), etc.

6.2.1 Clustering-based Anomaly Detection

In the clustering-based anomaly detection approaches, anomalies are generally seen as data samples that present a lower adhesion to the underlying groups. Several two-phase approaches have been proposed and consist in using a clustering algorithm such as DBSCAN (Sheridan et al.), K-means (Deng and Mei, 2009) and Affinity Propagation (Marcos Alvarez et al., 2013), then compute an anomaly score from the obtained clustering partition. Similarly in (Mahadevan et al., 2010), to detect temporal anomalies in videos, inlier behaviors are modeled as a mixture of Gaussian distributions. A deep GMM-based approach called DAGMM is proposed in (Zong et al., 2018) to detect outliers in numerical data, where the input data are compressed into a lower-dimensional space using an autoencoder and then

fed into a GMM component. The autoencoder's reconstruction loss and the log-likelihood of the GMM component are optimized jointly, without performing any pre-training phase.

6.2.2 Anomaly Detection in Text Data

Unlike images, time series, and numerical data, relatively few anomaly detection studies are dedicated to textual data. Document-term matrix representations (also called sparse bag-of-words) have previously been used in (Kannan et al., 2017) to perform anomaly detection based on Nonnegative Matrix Factorisation (NMF) and isolate an outlier matrix, used to compute the anomaly scores. Sparse representations are also used by Manevitz and Yousef (2001) as input to a One-Class Support Vector Machine (OC-SVM) (Schölkopf et al., 2001) and later to a shallow autoencoder (Manevitz and Yousef, 2007). Word embeddings like word2vec are used for anomaly detection in (Zhuang et al., 2017) along with a von Mises Fisher (vMF) mixture model where more general words are penalized when computing the overall outlierness score of a given document. Pre-trained fastText word vectors are used in (Ruff et al., 2019) as the embedding layer of a multi-head attention network to perform anomaly detection as a one-class classification task. Recently, a deep end-to-end approach has been proposed by Manolache et al. (2021) that does not use any knowledge transfer. The authors used the transformer architecture of ELECTRA (Clark et al., 2020) that contains two adversarial components: a generator and a discriminator. The model is trained from scratch on a given dataset by optimizing a loss function based on token replacement.

6.2.3 Semantic Text Representations

Tremendous advances in various NLP tasks have been made in recent years thanks to dense vector representations of words and text sequences. Static word embeddings like word2vec (Mikolov et al., 2013b), GloVe (Pennington et al., 2014) and fastText (Bojanowski et al., 2017) provide one unique dense representation for each word whereas contextual word embedding models like ELMo (Peters et al., 2018b), BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019b) provide word representations that depend on the surrounding context. Contextual word embedding models are based on deep neural networks, which makes them resource-intensive and difficult to use in some industrial contexts. Both kinds of word embeddings have proved effective in several unsupervised downstream tasks like semantic textual similarity (Arora et al., 2017; Ranasinghe et al., 2019), clustering (Boutalbi et al., 2022) and anomaly detection (Zhuang et al., 2017).

6.3 Gaussian Mixture Models

Given a corpus $\mathcal{D}$ of n short texts $(d_1, \ldots, d_n)$, we represent each sample by a fixed size vector, thus obtaining a matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ of size $n \times m$. To tackle the anomaly detection problem, we postulate that the samples follow a mixture of distributions, from which the anomalous samples deviate.

Admittedly, the family of t-distributions provides a heavy-tailed alternative to the Gaussian family for anomaly detection. However, as pointed out by Yuan and Huang (2009), although useful from modeling perspective, the practical use of multivariate t-distribution is often limited by the difficulty in parameter estimation, particularly so for high dimensional data. Note that, in our proposal, the consideration of the time consumption is important. Therefore, considering a t mixture model leads to estimate a supplementary parameter (in addition to the estimation of vector means and covariance matrices) that is the degree of freedom of each component. Moreover, since we suggest to consider an ensemble method allowing to combine results by varying the number of components (cf. Section 6.3.1), we would therefore increase yet the computation time for estimation of the parameters. However, Gaussian Mixture Model (GMM)-based approaches, more parsimonious than t-mixture model, have shown their effectiveness in anomaly detection, such as DAGMM (Zong et al., 2018). For these reasons we retain GMM to address our purpose.

In a finite GMM, the data $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ are taken to constitute a sample of n independent instances of a random variable $\mathbf{X}$ in $\mathbb{R}^m$. Density can be expressed as:

$$f(\mathbf{x}_i; \Theta) = \sum_{k=1}^{g} \pi_k \varphi_k(\mathbf{x}_i | \mu_k, \Sigma_k), \forall i \in \{1, \ldots, n\}$$

where $\Theta = (\pi_1, \ldots, \pi_g, \mu_1, \ldots, \mu_g, \Sigma_1, \ldots, \Sigma_g)$, $\varphi_k(\mathbf{x}_i | \mu, \Sigma_k)$ is the kth component density for observation $\mathbf{x}_i$ with parameters (μ_k, Σ_k), $(\pi_1, \ldots, \pi_{g-1})$ are the mixing weights or probabilities (such that $\pi_k > 0$, $\sum_{k=1}^{g} \pi_k = 1$) and g is the number of mixture components. Thus, clusters are ellipsoidal, centered at the mean vector μ_k, and with other geometric features, such as volume, shape and orientation, determined by the covariance matrix Σ_k (Banfield and Raftery, 1993; Celeux and Govaert, 1995). To estimate Θ we rely on the maximisation of the log-likelihood given by:

$$L(\mathbf{X}; \Theta) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{g} \pi_k \varphi_k(\mathbf{x}_i | \mu_k, \Sigma_k) \right).$$

The maximization is commonly performed by Expectation-Maximization (Dempster et al., 1977); an iterative algorithm based on the maximization of the conditional expectation of

the complete data log-likelihood given Θ':

$$Q(\Theta|\Theta') = \sum_i \sum_k s_{ik} \log(\pi_k \varphi_k(\mathbf{x}_i|\mu_k, \Sigma_k))$$

where $s_{ik} \propto \pi_k \varphi_\ell(\mathbf{x}_i|\mu_k, \Sigma_k)$ are the posterior probabilities.

In real terms, the algorithm is broken down into two steps (E-M steps) and the unknown parameters of Θ are updated thanks to the previously computed probabilities. For each component k, we have

$$\pi_k = \frac{\sum_i s_{ik}}{n} \quad \mu_k = \frac{\sum_i s_{ik} x_{ij}}{\sum_i s_{ik}}$$

$$\text{and } \Sigma_k = \frac{\sum_i s_{ik} (\mathbf{x}_i - \mu_k)^\top (\mathbf{x}_i - \mu_k)}{\sum_i s_{ik}}.$$

The procedure used to identify anomalies is described in Algorithm 5. It takes as input a

Algorithm 5: Unsupervised anomaly detection with GMM

Input: $\mathcal{D} = \{d_1, \dots d_n\}$, g the number of components, $\mathcal{M}$ an embedding module, α the desired number of output samples;

$\mathbf{x}_i \leftarrow \mathcal{M}(d_i)$, $i = 1, \dots n$;
$\mathbf{X} \leftarrow (\mathbf{x}_1, \dots, \mathbf{x}_n)$;
Initialize Θ' from a partition obtained with k-means
repeat
    E-step: Compute $Q(\Theta|\Theta')$;
    M-step: Update π_k, μ_k and Σ_k;
until *Convergence*;
$s_i \leftarrow -\max_k(s_{ik}), k = 1, \dots, g$;
$\mathbf{s} \leftarrow (s_i, \dots, s_n)$;
$\mathbf{r} \leftarrow$argsort$(\mathbf{s})$;
return d_j, $j = \mathbf{r}_1, \dots, \mathbf{r}_\alpha$;

set of short texts and returns the ones that are the most likely to constitute an anomaly. The maximum density as normality score denotes the confidence of the assignment. Multiplied by -1, it denotes the uncertainty of the assignment and is similar to using the entropy of $\mathbf{s}$ since $\sum_k s_{ik} = 1$. The number of returned text samples depends on the user's needs and is specified by the cutoff parameter α. In the evaluation section, we evaluate the anomaly detection performance with every possible value of α using the AUROC score.

6.3.1 Proposed Solution for Multi-Class Inliers

In the standard setting of anomaly detection where we consider one large inlier class, we set the number of components to its smallest possible value $g = 2$, which provides satisfactory results. In this study, we also consider the more challenging scenario where several underlying topics are present in the dataset. In this context, we make the distinction between extreme values and outliers (Aggarwal, 2017) as shown in Figure 6.2. A Gaussian mixture model would not have any difficulty in spotting both types of outliers since it is capable of modeling clusters of different shapes. Furthermore, we expect GMM to show good results in the multi-class context, since one of its fundamental assumptions is the multiplicity of inherent distributions among the data samples. However, this property requires to know the number of components in advance, which is not always possible in real life. To address this issue, we propose to use GMME, an ensemble of several models, obtained with different values of g. To this end, we use Algorithm 5 with varying $g_k \in \mathcal{G}$ and combine the output scores $\mathbf{s}^{(g_k)}$ as follows:

$$e_i = -\frac{1}{|\mathcal{G}|} \sum_k \text{rank}(s_i^{(g_k)}), \ \forall i = 1, \ldots, n. \tag{6.1}$$

The intuition behind using an ensemble approach is to make each of the models separate the dataset into clusters in a different way and assign an anomaly score according to the formed clusters. Combining those different anomaly scores leads to a more robust and meaningful overall score, even when the optimal number of clusters is not included in $\mathcal{G}$. This is corroborated by the empirical study conducted in Section 6.4.5.

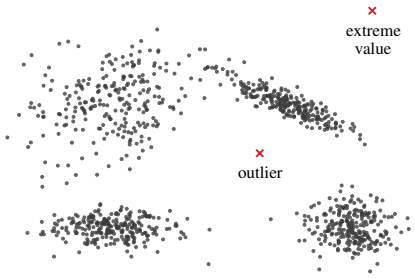

Figure 6.2: Difference between outlier and extreme value. This example illustrates the benefit of the clustering-based anomaly detection approaches in general and Gaussian mixture models in particular.

6.4 Experimental Study

To assess the effectiveness of our approach and compare it to state-of-the-art, we conduct a set of experiments on real datasets and discuss the results in this section.

6.4.1 Datasets

We run our anomaly detection experiments on three datasets described in Table 6.2. MLSUM (Scialom et al., 2020) and COVID-news (Cortal, 2022) are both news datasets from which we extract the title to constitute our short-text corpora. RNCP is a dataset we built from an official french repository that lists training certifications.

Table 6.2: Datasets' description. The sizes correspond to the whole set of samples (training and test set).

Dataset	Classes	Smallest	Largest	Medial
RNCP	16	763	9,510	3,101
COVID	9	236	3,235	1,270
MLSUM	10	2,573	26,024	13,054

Given a classification dataset, we first remove the classes that are too small to constitute an anomaly detection dataset, thus obtaining ℓ classes. We then derive ℓ sets of samples in which there is one inlier (majority) class and a certain rate r of outliers picked randomly from the other classes.

Construction of the RNCP dataset

The RNCP (*Répertoire National des Certifications Professionnelles*) dataset is composed of certification contents that are provided by an organization named « *France-Compétences* ». All the data used here are publicly available[2].

In order to label data we first get the ROME codes of each certification that correspond to the professions related to the certification. Each ROME code is assigned to a topic in the file « Arborescence thématique » of the same repository. We use the ROME code as intermediate to have the topic (*thématique*) of each certification. We hence obtain a multi-label classification dataset.

[2] https://www.data.gouv.fr/fr/datasets/

Given a certification c_i, let $\mathcal{T}_i$ be the set of its corresponding topics. In the one-class setting with inlier class h, the label of c_i is established as follows:

$$y_i = \begin{cases} 0 & \text{if } h \in \mathcal{T}_i \\ 1 & \text{otherwise.} \end{cases}$$

In the multi-class setting with inlier classes symbolized by $\mathcal{H}$, the anomaly labels are defined as:

$$y_i = \begin{cases} 0 & \text{if } \mathcal{H} \subset \mathcal{T}_i \\ 1 & \text{if } \mathcal{H} \cap \mathcal{T}_i = \emptyset \\ \text{N/A} & \text{otherwise} \end{cases}$$

where $y_i = 1$ means c_i is categorized as an anomaly.

6.4.2 Experimental Settings

In order to empirically evaluate our approach and compare it to other anomaly detection techniques, we rely on the Area Under the Receiver Operating Curve (AUROC), originally used as a metric in the classification task. In our case, it takes as input the anomaly scores as well as the ground truth labels and determines at what extent it is possible to accurately identify outliers using the anomaly score. It is equivalent to evaluating the performance of anomaly detection at every possible value of α in Algorithm 5. For each anomaly detection approach, we compute the AUROC on the test set over 5 different initializations, except for OC-SVM that is a deterministic model.

In our study, we discard the case where we train the model on the majority class as a one-class classification task (Manevitz and Yousef, 2001; Manolache et al., 2021; Ruff et al., 2019), for it is not a realistic scenario. Indeed, in the context of anomaly detection, in contrast to novelty detection for example, one rarely has access to a large enough amount of inlier-only labeled samples in real life. Thus, we consider in this study a fully unsupervised scenario, where no labels are available and both inlier and outlier samples are present in the training set. To this end, we contaminate both sets with up to $r = 10\%$ of outliers as in (Manolache et al., 2021). The set of labels is used only during the evaluation phase.

Text representation. Given a raw corpus $\mathcal{D}$ of short texts, we first perform a minimal pre-processing that consists in removing stop words and lowercasing the input text. Then, we use a pre-trained fastText model (Bojanowski et al., 2017) to represent texts by fixed-size vectors and build the $\mathbf{X}$ matrix. The model is trained on French Wikipedia and represents

each word by a vector of size $m = 300$. Using those word vectors, we represent a text sequence by the arithmetic mean of its tokens' representations as in (Arora et al., 2017; Ranasinghe et al., 2019). We show that this way of representing text sequences is well suited to shorts texts and is very beneficial in capturing text semantics for anomaly detection. One noticeable advantage of fastText is its ability to represent out-of-vocabulary words thanks to sub-word embeddings. Another advantage of fastText is that pre-trained word representations are provided in a wide range of languages[3]. We do not use contextual word embeddings to represent text sequences since they significantly increase the computational cost and do not seem to bring any performance gain in the anomaly detection task (Ruff et al., 2019).

Baselines. We compare our approach to other anomaly detection techniques: OC-SVM (Schölkopf et al., 2001), AE, DAGMM (Zong et al., 2018) and DATE (Manolache et al., 2021). For OC-SVM, AE and DAGMM, we use as input the same matrix $\mathbf{X}$ as for GMM. Concerning OC-SVM, we set $\nu = 0.05$, which is the value that presents the best results among $\{0.05, 0.1, 0.2, 0.5\}$ by far. For the autoencoder (AE) we train a model with three encoder layers and three decoder layers of size 256, 128 and 64, a learning rate of 0.001 and a weight decay of 10^{-8}. Concerning DAGMM, the authors chose the parameters relating to the architecture of the neural network according to the dataset and do not provide a method to reproduce this choice. This way of configuring the model is not suitable for the unsupervised case, in which no tuning of the hyperparameters is possible. We therefore opt for a standard architecture in decreasing powers of 2 starting from $m = 300$ (i.e. $256, 128, \ldots$). We choose $m' = 5$ as the encoding dimension because it gives the best overall performance. For DATE, we use the same parameters as used in the original paper for AG-news (Manolache et al., 2021).

Hyperparameter tuning. Our present work falls within the context of unsupervised learning, where we're not supposed to have access to ground truth labels. In this respect, as discussed in Section 1.5, it is not feasible in real life to tune the hyperparameters of a given model to fit unlabeled data, since the performance score is simply impossible to compute. We therefore consider hyperparameter tuning in this context unrealistic and possibly leading to hiding instabilities that can neither be detected nor fixed by practitioners. Hence, robustness and insensitivity to parameters are key in the unsupervised setting. For this reason, regarding the baselines, we use the recommended parameters provided in the original papers when available, in order to reproduce real-world conditions. When

[3] https://fasttext.cc/

no recommended parameters are provided, we use the ones that maximize the overall performance even though it does not play in favor of our proposed approach which does not require any parameter tuning. By doing so, we guarantee a fair evaluation of our proposal and show its robustness in the unsupervised context we address in this chapter.

6.4.3 Results with One-Class Inliers (classical setting)

The obtained performance is given in Table 6.3 with an anomaly rate $r = 10\%$. We first observe the effectiveness of GMM on the three datasets, in comparison to all of the baselines, offering the best AUROC in most of the cases. Furthermore, the state-of-the-art DATE shows its limits on short texts and presents competitive but poorer results in comparison to GMM. OC-SVM is competitive on short texts in comparison to DATE but presents poorer overall performance. AE is the model that provides the lowest AUROC values, right after DAGMM. This might be due to the fact that the encoding (or embedding) step of the encoder (for both AE and DAGMM) is performed beforehand using pre-trained word embeddings and becomes pointless when applied to this kind of representation.

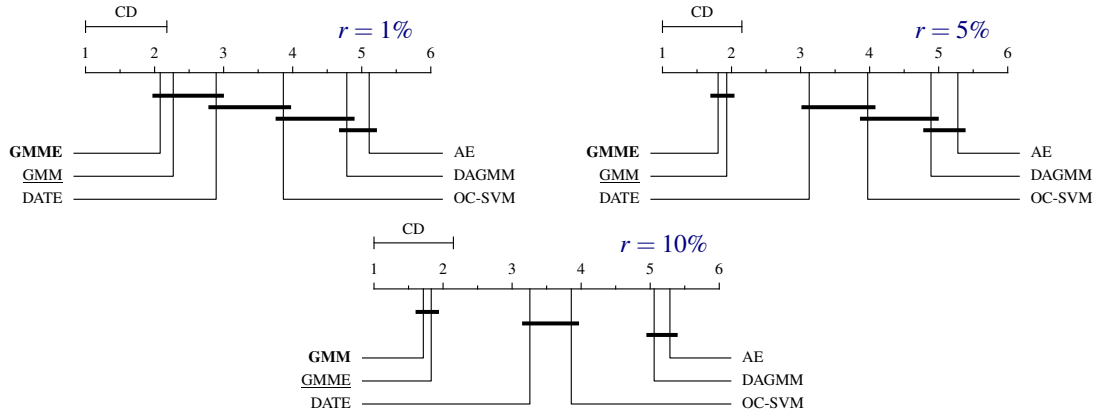

Figure 6.3: CD plots from the Nemenyi test over different datasets. This graphic summarizes the rank of each approach with different contamination rates r.

The results obtained with different values of contamination rate r are summarized by Critical Difference (CD) diagrams in Figure 6.3. The aim of CD diagrams (Demšar, 2006) is to visualize the performance ranks of each approach over the different datasets. If we take the example of $r = 10\%$, the CD diagram summarizes the scores given in Table 6.3. It depicts the average rank of each method and the bold line corresponds to the critical difference, based on the post-hoc Nemenyi test (Nemenyi, 1963). Note that the presence of a higher number of outliers increases the difficulty of the anomaly detection task and decreases

Table 6.3: AUC scores obtained with anomaly $r = 10\%$. The **bold** numbers correspond to the best score in each row and the underlined numbers are for the second-best performance score.

	Inlier	OC-SVM	AE	DAGMM	DATE	GMM
RNCP	environ.	0.622	0.621	0.528	0.658	**0.684**
	défense	0.580	0.462	0.525	0.622	**0.743**
	intelli.	0.667	0.654	0.591	0.790	**0.801**
	recherc.	0.686	0.645	0.580	0.753	**0.777**
	nautism.	0.649	0.629	0.530	0.682	**0.731**
	aéronot.	0.637	0.594	0.498	0.761	**0.780**
	sécurit.	0.598	0.547	0.513	0.750	**0.800**
	multimé.	0.578	0.563	0.513	0.645	**0.712**
	humanit.	0.567	0.596	0.534	0.690	**0.721**
	nucléai.	0.651	0.587	0.599	0.749	**0.751**
	enfance	0.610	0.601	0.650	0.721	**0.789**
	saisonn.	0.505	0.475	0.508	0.488	**0.749**
	assista.	0.451	0.507	0.564	0.525	**0.632**
	sport	0.511	0.561	0.468	0.663	**0.733**
	ingénie.	0.669	0.601	0.572	**0.753**	0.742
	sans di.	0.454	0.369	0.537	0.393	**0.599**
COVID	culture.	0.497	0.401	0.434	0.403	**0.537**
	environ.	0.569	0.550	0.502	0.552	**0.662**
	interna.	0.620	0.537	0.531	0.624	**0.651**
	people	0.639	0.474	0.497	0.518	**0.640**
	politiq.	0.672	0.601	0.516	0.681	**0.792**
	science.	0.555	0.383	0.606	0.645	**0.666**
	société	0.489	0.475	0.553	**0.570**	0.551
	sport	0.685	0.356	0.554	0.526	**0.690**
	économi.	0.527	0.494	0.507	0.540	**0.594**
MLSUM	afrique	0.700	0.505	0.620	0.743	**0.755**
	police-.	0.783	0.744	0.529	0.770	**0.829**
	politiq.	0.703	0.603	0.496	0.746	**0.757**
	livres	**0.712**	0.658	0.484	0.581	0.659
	argent	0.477	0.364	0.517	**0.822**	0.634
	culture	**0.595**	0.498	0.524	0.458	0.514
	sante	0.787	0.562	0.552	0.865	**0.875**
	science.	0.531	0.355	0.496	**0.710**	0.552
	societe	0.730	0.730	0.489	0.754	**0.790**
	sport	0.749	0.565	0.452	0.692	**0.817**

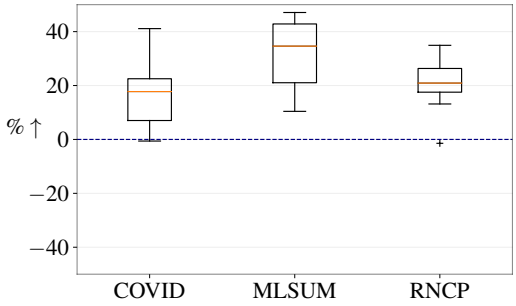

Figure 6.4: Gain of performance between GMM and VMF-Q using word2vec embeddings of size $m = 200$. Positive values give advantage to GMM.

the overall performance scores. We can observe that the more we inject anomalies in the training set, the more GMM gets competitive against the other approaches in terms of AUROC score. This shows GMM's robustness to outliers and its generalization capabilities while other techniques tend to overfit in the presence of noise in the training set.

Also, it is worth noting that GMME yields similar results in comparison to GMM, which makes it a universal solution, well suited to anomaly detection even in the one-class scenario.

Comparison with VMF-Q. To make a fair comparison between GMM and VMF-Q (Zhuang et al., 2017), we reproduce the exact same setting for the two approaches. VMF-Q is based on the von Mises-Fisher distribution that relies on a Bessel function to estimate the parameter κ. The model is originally trained using embeddings of size $m = 200$, but encounters numerical difficulties with higher dimensions, due to the approximations made by the Bessel function that depends, inter alia, on m. We hence use, for both GMM and VMF-Q, another pre-trained model of size $m = 200$, that is provided by Fauconnier (2015). The gain of performance from VMF-Q to GMM is summarized in Figure 6.4. We can observe a clear advantage of GMM in comparison to VMF-Q on the three datasets, especially on MLSUM, where GMM outperforms VMF-Q on all the subsets. We also report poorer overall results using word2vec with $m = 200$ in comparison to the fastText model we use in the rest of our experiments.

6.4.4 Performance and Data Size

Figure 6.5 shows the gain of performance from baselines to GMM according to the size of the datasets. We see that the percentage of improvement is greater on small datasets but remains positive on large datasets. Note that detecting anomalies can be trickier on small

datasets, especially when dealing with short texts (cf. Table 6.1) which makes GMM a good solution to tackle this difficulty.

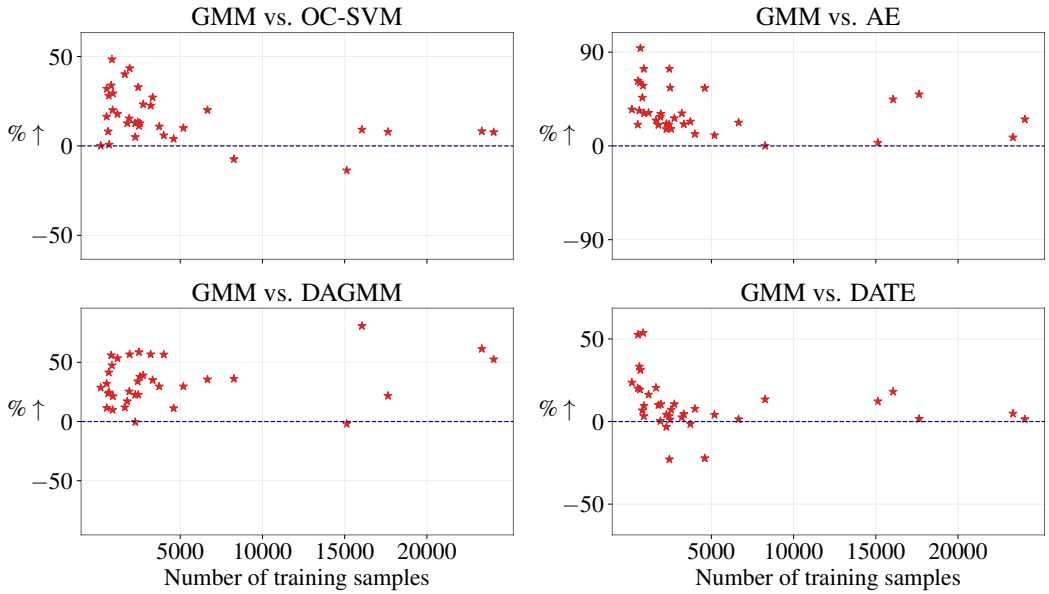

Figure 6.5: Percentage of improvement of GMM in comparison to baselines w.r.t. to the train set's size. Positive values give advantage to GMM.

6.4.5 Multi-Class Inliers

We investigated in the previous sections the detection of semantic anomalies in a dataset classically composed of one unique class. In this section, we consider a dataset with several underlying topics and identify the samples that do not belong to any of them. To evaluate our approach in such a context, we create datasets as described in Section 6.4.1 but this time, by combining $\hat{g}$ random classes to form one inlier class then inject anomalous samples with a rate $r = 10\%$. We then proceed similarly to identify anomalies and validate the obtained results. Note that we make sure in our experiments to put aside enough "anomalous" classes so that we have a sufficient diversity of anomalies and avoid forming an additional cluster with the anomalous samples. To this end, we limit $\hat{g}$ according to the available classes in the dataset. We use the obtained datasets to assess the performance of GMME (cf. Section 6.3) and compare it to GMM and two other baselines: DBSCAN and DATE. DBSCAN natively deals with outliers and considers the samples that cannot be assigned to any existing cluster as anomalies (they are assigned the -1 label). We use this information to determine whether a text sample is an anomaly. We set the parameters $\epsilon = 1$ and min_samples $= 3$ which yielded better results than the default values available in the scikit-learn implementation.

Table 6.4: Comparison of DBSCAN, DATE, GMM and GMME approaches: AUROC scores obtained with multiple classes as inliers.

Dataset	# inlier classes	DBSCAN	DATE	GMM	GMME
RNCP	2	0.634	0.756	0.886	**0.892**
	3	0.628	0.705	0.824	**0.839**
	4	0.646	0.776	0.802	**0.82**
	5	0.657	0.635	**0.816**	0.807
	6	0.617	0.567	**0.655**	0.645
	7	0.584	0.488	0.606	**0.626**
	8	0.604	0.568	0.632	**0.634**
COVID	2	0.515	0.593	**0.652**	0.638
	3	0.506	0.513	0.541	**0.542**
	4	0.507	0.535	0.562	**0.563**
	5	0.5	0.511	0.526	**0.527**
MLSUM	2	0.505	0.7	0.742	**0.746**
	3	0.502	0.496	**0.551**	0.544
	4	0.506	0.47	0.55	**0.555**
	5	0.507	0.457	0.531	**0.538**

Table 6.4 shows the performance obtained by GMME with values of $g_k \in \mathcal{G} = \{2, 3, 4\}$. We first observe that the ensemble approach further improves the performance of the simple GMM, which still performs better in comparison to the other baselines. DBSCAN presents the poorest results after DATE, which is less competitive in the multi-class context, especially on the RNCP dataset. Thus, GMME is the most competitive approach in the multi-topic scenario, even when the real number of clusters is not included in $\mathcal{G}$.

6.4.6 Computation Time Analysis

Table 6.5 contains the execution time of both the training and evaluation steps. It is estimated over three different runs, on the three-class datasets used in Section 6.4.5. The sizes of the training sets are 1034, 6342, 46253 and the test sets are of size 444, 2718, 5837 for RNCP, COVID-news and MLSUM respectively. The experiments on DATE are performed on an NVIDIA RTX2070 GPU.

We first notice that GMM is the fastest approach both during the training and evaluation phase. It is followed by GMME that is relatively quick, especially during evaluation. GMM scales better with an increasing number of samples in comparison to OC-SVM that takes more than four times as much time to train. DATE is the approach that takes the most

Table 6.5: Execution time in seconds. The row $\mathbf{X}$ corresponds to the computation time of the embedding matrix $\mathbf{X}$ using fastText. “train“ stands for the training time and “eval“ for the evaluation time.

Dataset	Step	1-SVM	GMM	GMME	DATE
RNCP	$\mathbf{X}$	3.7			-
	training	3e-01	3e-01	1.2	78.49
	evaluation	1e-01	3e-03	3e-02	2.0
COVID	$\mathbf{X}$	3.7			-
	training	5.7	3.5	16.2	512.97
	evaluation	1.8	2e-02	8e-02	14.1
MLSUM	$\mathbf{X}$	3.9			-
	training	450.6	10.0	94.8	3639
	evaluation	33.0	4e-02	2e-01	30.2

time to train and evaluate, which is due to its deep architecture. The computational time of DATE can be partially amortized with a more powerful GPU but can still represent an impediment, especially in an industrial context.

We also observe that the computation time of $\mathbf{X}$ does not depend much on the size of the dataset, simply because the task that takes the longest is loading the model from memory. Hence, this way of representing text scales well, especially when used along with GMM or GMME.

6.5 Anomaly Examples with RNCP

Table 6.6 presents some examples of anomalies predicted on two subsets of the RNCP dataset: « *aéronotique* » (meaning aeronautics) and « *nucléaire* » (meaning nuclear). The *aéronautique* test set contains 1585 samples, 154 of which are labeled as anomalies, and the *nucléaire* set contains 1431 samples including 131 anomalies. In both cases we set α to 250 (cf. Algorithm 5). We observe in both cases that DATE has more difficulty in detecting anomalous text sequences when they are very short. For example, in the *nucléaire* set, the certification *Livreur* (meaning delivery person) does not have anything to do with the nuclear field. Yet DATE does not place it among the 250 most deviant samples and makes it the 355th anomalous sample while it is only 3rd according to GMM. This might be explained by the fact that DATE is trained from scratch and does not benefit from the semantic knowledge inherited by transfer learning.

Table 6.6: Examples illustrating the difference of prediction between GMM and DATE according to the length of the text sequence, with $\alpha = 250$.

Subset	Certification	GMM	DATE	Real
aéronotique	Sciences, Technologies, Santé - Mention : Automatique et informatique industrielle - Spécialité : Automatismes industriels	Inlier	Inlier	Inlier
	Production industrielle option ingénierie des matériaux nouveaux	Inlier	Inlier	Inlier
	Actuaire	Outlier	Inlier	Outlier
	Sciences Politiques	Outlier	Inlier	Outlier
	CQP Animateur de patinoire option hockey sur glace	Outlier	Outlier	Outlier
	Décor architectural opt. B Domaine du décor du mur	Outlier	Outlier	Outlier
nucléaire	Culture et communication Mention : Création, innovation, information numériques Spécialité : Gestion de l'information et du document Domaine : Culture et communication	Inlier	Inlier	Inlier
	Responsable d'ingénierie des systèmes d'information et de communication, option "analyse et développement", option "systèmes et réseaux" et option "télécommunications"	Inlier	Inlier	Inlier
	Livreur	Outlier	Inlier	Outlier
	Architecte d'intérieur	Outlier	Inlier	Outlier
	Responsable conception, mise en place et maintenance des installations frigorifiques et climatiques	Outlier	Outlier	Outlier
	Urbanisme et Aménagement Spécialité DYATER (Dynamiques et Aménagement des espaces, Territorialités)	Outlier	Outlier	Outlier

6.6 Improving the Results with Transformers

In the previous experiments, we relied on fastText static embeddings in order to show that is it possible to perform effective and fast anomaly detection without having access to important computational resources. However, it is possible to achieve even better results using Transformer representations. We use for our experiments CamemBERT (Martin et al., 2020) and FlauBERT (Le et al., 2020), both trained on French corpora.

Table 6.7: Description of the French Transformer models used for anomaly detection.

Model	Layers	Dimensions	Tokens	Training data
CamemBERT-base	12	768	32k	138 GB
CamemBERT-large	24	1,024		
FlauBERT-small	6	512	50k	71 GB
FlauBERT-base	12	768		
FlauBERT-large	24	1,024		

Table 6.8: Comparison between different embedding representations in terms of AUC score with anomaly rate $r = 10\%$. The **bold** numbers correspond to the best score in each row and the underlined numbers are for the second best performance score. flauB and camB stand respectively for FlauBERT and CamemBERT and the subscript is for the number of layers, which indicates the model size (6 for small, 12 for base and 24 for large).

Dataset	# Inlier classes	DATE	GMME					
			fastText	flauB_6	flauB_{12}	flauB_{24}	camB_{12}	camB_{24}
RNCP	2	0.756	**0.892**	0.854	0.644	0.797	0.838	0.846
	3	0.705	**0.839**	0.746	0.703	0.689	0.782	0.798
	4	0.776	0.820	0.785	0.751	0.702	**0.831**	0.818
	5	0.635	0.807	0.802	0.596	0.745	0.800	**0.816**
	6	0.567	0.645	0.610	0.571	0.639	0.665	**0.666**
	7	0.488	0.626	0.657	0.704	0.692	0.693	**0.722**
	8	0.568	0.634	0.538	0.512	0.621	**0.665**	**0.665**
COVID	2	0.593	**0.638**	0.600	0.540	0.595	0.612	0.561
	3	0.513	0.542	0.547	0.481	0.556	**0.563**	0.539
	4	0.535	0.563	0.557	0.513	0.532	**0.575**	0.556
	5	0.511	0.527	0.552	0.496	0.515	**0.555**	0.530
MLSUM	2	0.700	0.746	0.697	0.646	0.623	0.763	**0.809**
	3	0.496	0.544	0.611	0.495	0.579	0.609	**0.629**
	4	0.470	0.555	0.608	0.405	0.582	0.616	**0.619**
	5	0.457	0.538	0.608	0.432	0.581	0.613	**0.637**

CamemBERT and FlauBERT are both based on the RoBERTa variant of Transformer language models (Liu et al., 2019b). While FlauBERT uses the exact same objective and tokenization process as in (Liu et al., 2019b), plus a French-specific pre-processing, CamemBERT makes use of the SentencePiece tokenizer (Kudo and Richardson, 2018) and the whole-word masking strategy that consists in masking words instead of sub-word tokens. The major difference between the two models is the training set of data. CamemBERT uses the French part of OSCAR, a large multilingual corpus extracted from Common Crawl[4], while FlauBERT is trained on a set of 24 corpora. More information about the two models are given in Table 6.7.

Table 6.8 contains the results obtained using several input representations. We recall the results obtained by DATE for comparison purposes since it is also based on a Transformer architecture. We first observe the significant difference in performance between CamemBERT and FlauBERT with a clear advantage for CamemBERT in both its base and large

[4] https://commoncrawl.org/

versions. The large variant of CamemBERT achieves the best overall results, especially on MLSUM. CamemBERT-base yields competitive results, achieving the best AUROC scores on the COVID dataset. We also notice that, in all cases, the two-phased approach that consists in computing Transformer representations and then use an ensemble of model-based clustering models (GMME) is more efficient than the end-to-end Transformer-based approach (DATE). It is worth noting that fastText embeddings remain competitive and surpass the three FlauBERT models. It is therefore a very good alternative in real-time use cases when computational speed is a critical issue.

6.7 Industrial Application

In order to assess the effectiveness of the anomaly detection process in a real world situation where no labels are available, we apply GMME using fastText on an unlabeled dataset provided by the service « *Direction de la Formation Professionnelle* » (DFP) for qualitative evaluation purposes. This dataset corresponds to a particular category of training courses called « *Actions de formation dispensées aux créateurs et repreneurs d'entreprise* » (ACRE) and is supposed to contain trainings that are targeted for the users who are involved in the acquisition of a business. The motivation for the choice of this set of data are as follows:

- The ACRE category is known by the experts of the DFP for being particularly prone to anomalies. This is explained by the fact that training providers are obliged to specify the category to which the training belongs (cf. Figure 6.1) since the courses that do not belong to any pre-determined category are considered non-eligible. This leads some training providers to specifying a default category to their training, and it has been observed that their first choice is usually the ACRE category, which makes it an absolute **priority** within the anomaly detection objective.
- The presence of a high number of anomalies makes the task of outlier detection more **challenging**, since it becomes more difficult to differentiate inlier classes and anomalies.
- ACRE is a relatively broad category in which several **sub-topics** can be present such as accounting and legal issues, which makes the GMME approach more suitable.

To respect the real-time perspective, we use the same experimental setting described previously, which consists in training the model on a training set and evaluate it on a test set. Also, in order to better simulate the real-world configuration, we choose the latest samples as the test set. Specifically, we considered as the training set all the trainings specified

Table 6.9: Anomaly detection examples on a real-world unsupervised dataset with GMME using fastText.

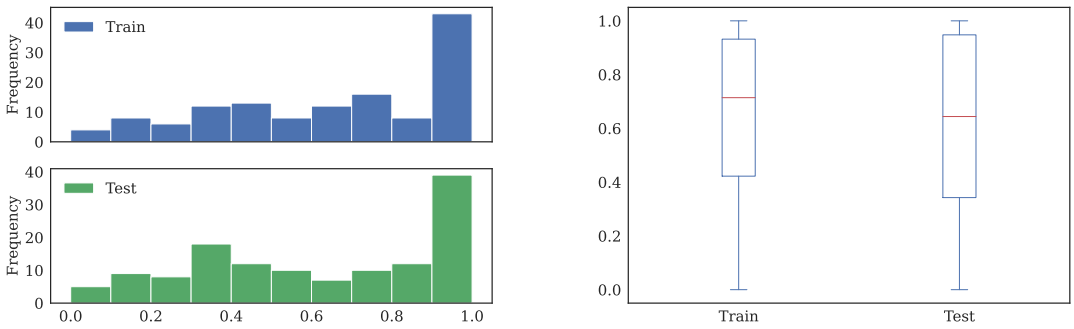

Figure 6.6: Distribution of the normalized anomaly score obtained with GMME on the ACRE category.

as bellonging to the ACRE category and created before November 2022, which results in 13,293 observations. For the test set, we take the trainings created since November 2022, resulting in 669 samples. The degree of abnormality is computed using equation 6.1, after a `minmax` normalization which consists in using $\frac{e_i - e_{min}}{e_{min} - e_{max}}$, thus producing values between 0 and 1 without altering the original shape of data. The distribution in both the training and test set are given in Figure 6.6 in which we observe a similar distribution between the two sets of data, with a relatively high number of values between 0.9 and 1 as well as a slightly lower median in the test set.

In addition, examples with different degrees of abnormality are provided in Table 6.9. We first observe that the more the anomaly score gets higher, the more we veer off the subject of « *création d'entreprise* » both in the training and thet test set. For example, moderate anomalies include « *Les bases de la gestion du temps* » and « *Prévenir et gérer les conflits interpersonnels* » in the train set and « *Créer votre entreprise en soins esthétiques hautes technologies Microblading* » and « *Développer mon activité dans les ongle* » in the test set, all of which are too specific to be considered eligible. In order to qualitatively evaluate the precision of the model, i.e. the relevance of the anomalies detected, we can analyse the highest anomaly scores such as « *Formation HTML / CSS* » and « *Transformation du porc* » in the train set and « *Massage aux pierres chaudes* » and « *Les fondamentaux de la lutte contre les maladies de la Vigne* » in the test set, all of which are clearly irrelevant and are likely to be deliberately put in the ACRE category. On the other hand, example that are likely to constitute unintentional mistakes are « *Bilan de compétences* », « *Apprentissage des langue* » since categories corresponding to these training contents are available, which makes them eligible trainings. Both scenarios are, of course, targeted by the anomaly detection system and require either the correction or the withdrawal of the anomalous training session.

6.8 Conclusion

This chapter addresses semantic anomaly detection in short texts with an additional constraint in time efficiency. In addition to the classical framework where one class is used as the inlier class, we also consider the scenario where several underlying subgroups are present in the normal class. We see anomaly detection as a probabilistic clustering problem, in which we learn a Gaussian mixture model and consider the low posterior probability samples as belonging to none of the modeled clusters and more likely to constitute outliers. This uncertainty score proved effective with different numbers of subgroups. In the multi-class setting, we propose GMME, an ensemble approach that improves the performance of GMM when several topics are present in the inlier class. The two approaches outperform state-of-the-art anomaly detection techniques in both scenarios, with an impressively low computation time.

In our proposal, we rely on the Gaussian Mixture model for its flexibility. This choice is motivated by the presence of the proportions π_k of each cluster and the spectral decomposition of the covariance matrix Σ_k taking into account the volume, shape, and orientation of each cluster (as depicted in Figure 6.2). The characteristics of the clusters should not be overlooked when tackling the problem of anomaly detection through a clustering approach. Furthermore, note that our approach can be extended to latent block models, devoted to co-clustering, which may constitute an interesting and promising future path of research.

General Conclusion and Perspectives

This thesis has been dedicated to extending the knowledge about the use of neural text representations in the unsupervised domain of machine learning, with applications to clustering, visualization and anomaly detection. The studies conducted throughout this work are intended to respect as much as possible the constraints of unsupervised learning, in which very few elements are known a priori about the data. The key results and findings of this thesis can be summarized as follows:

- We first contribute to enriching the knowledge about the black-box multi-layered Transformer models, by analyzing the behavior of each layer. In Chapter 2, we conduct an unsupervised study in order to compare layers with each other and establish similarities as well as discrepancies across layers and between different Transformer models. Our study allowed us to identify groups of layers that present similar behaviors both in terms of external statistical measures and quality. In Chapter 3, we investigate the complementarity between the different layers of the same Transformer model, by combining them into one common data representation, obtained using multiway techniques. These methods free us from the need of choosing the right layer to perform clustering, while combining the useful information brought by each layer of the deep architecture. We observed through our empirical study the superiority of multiblock techniques, especially MFA, as opposed to tensor-based approaches in summarizing the useful information present in 24 layers into only 10 dimensions.

- In Chapter 4, we focus on document-level representations to perform document clustering. To this end, we propose an ensemble approach that successfully harnesses the whole set of layers in a fully unsupervised way. Combined with PCA-based dimension reduction along with a whitening operation, the results are further improved, achieving better results than the best-performing layer, which is impossible to determine in the absence of labels. Known for its robustness, the ensemble paradigm, as used in our approach, has shown significant improvement over simultaneous approaches. This might be explained by the fact that simultaneous approaches may observe a

certain influence between layers, while the ensemble approach, via its downstream combination of information, allows more freedom to each layer-wise partition. In order to take our unsupervised study even further and show its applicability, with satisfying results, in the case where the exact number of clusters is unknown. To this end, we take advantage of the ensemble approach by varying the number of clusters during the layer-wise clustering phase, which allows for achieving results that are comparable to the classical setting where the exact number of clusters is a priori known.

- In Chapter 5, we go further and investigate more dimension reduction techniques in order to further improve the quality of pre-trained text representations, as part of a tandem approach that combines dimension reduction and clustering. Among the evaluated approaches, UMAP stood out from the rest, showing impressive results while drastically reducing the dimensionality. More importantly, we show that the tandem approach significantly outperforms the fine-tuning strategies previously proposed in the literature and is supposed to be well-suited to any unsupervised downstream task. In this chapter, we also investigate the reasons behind such performance levels, by verifying if the improvements brought by the tandem approach are due to a decrease in the anisotropy of the embedding space. The anisotropy phenomenon, observed in several dense text representations, is characterized by narrow angles between embedding vectors, which has been suspected to reduce the expressiveness of pre-trained representations. By using the clustering performance as a measure of the quality of representations, we show that the impact of anisotropy has been overstated and that it does not have any noticeable effect on the capability of the document vectors representations.

- In Chapter 6, we address an industrial problem encountered in the platform *moncompteformation*, which is to spot training contents that are either non-eligible or wrongly categorized. Since we do not rely on any labeled data, this problem can be formulated as an unsupervised anomaly detection task in which we aim at identifying, given a category $\mathcal{C}_i$ of training contents, atypical instances that are less likely to truly belong to $\mathcal{C}_i$. More specifically, we intended to build a system that evaluates, in a short timeframe, the degree of abnormality of a set of short texts, containing an unknown number of sub-categories. To this end, we propose to use a set of Gaussian mixture models in order to identify semantically deviating samples. A fast approach that uses static word embeddings has shown substantial improvements in comparison to previously proposed approaches, both in terms of performance and time efficiency.

Further performance improvements have been observed using Transformer models, which constitutes a good alternative in the absence of time computation constraint.

Overall, the findings issued throughout this thesis highlighted the potential of pre-trained neural text representations in unsupervised text-mining tasks and may motivate further investigations such as:

- In our studies, we included several static and contextual models, thus observing different behavior patterns from one model to another. It is worthwhile considering other architectures like adversarial (Clark et al., 2020) and generative (Lewis et al., 2020; Raffel et al., 2020), that would potentially present new patterns and different performance levels in unsupervised tasks. Besides, other promising static embedding models may be used to perform unsupervised learning such as the directional model JoSE (Meng et al., 2019) which has the characteristic of producing embedding vectors that lie in the unit hypersphere. Hence, some co-clustering algorithms derived from appropriate von-Mises Fisher mixture models or latent block models proposed in (Affeldt et al., 2021; Govaert and Nadif, 2013; Salah and Nadif, 2017a, 2019) deserve to be tested on such embeddings.

- Regarding the clustering task, we exclusively investigated the use of pre-trained representations, which has led to promising results, provided a proper use of the raw vectors. An ambitious path for future work is to learn text representations that are specifically well suited to the clustering task. The objective of such models may be inspired by manifold dimension reduction techniques such as UMAP (McInnes et al., 2018) that handily combines local and global information to produce meaningful and well-separated clusters.

- Admittedly, our investigation of anomaly detection in short text data may be deepened and paths for future research are multiple. For instance, the undeniable potential of mixture models can be further investigated by making them even more robust to outliers and specifically designed for anomaly detection. An interesting idea proposed by Yu et al. (2015a) for the univariate setting consists in incorporating a sparse mean-shift penalization that preferentially pulls the outliers to the mean with the aim of performing robust parameter estimation and, simultaneously, identifying atypical samples in the set of observations. A promising perspective is to investigate the impact of such penalization on the multivariate version of mixtures models.

- In this work, we have shown a great benefit of the French pre-trained embedding models like fastText (Bojanowski et al., 2017) and Camembert (Martin et al., 2020) in

the general-purpose anomaly detection task. In order to better meet the industrial objective of spotting anomalies in training contents, it would be interesting to go further and adapt the models not only to the language but also to the education domain. Indeed, the vocabulary used in the course contents is very specific, and a specialized model should help distinguish subtle turns of phrases within the educational textual contents. To this end, it is possible to follow the process applied to other areas such as law (Chalkidis et al., 2020) and the clinical domain (Alsentzer et al., 2019), possibly by taking Camembert a starting point and retraining it on the data issued from *moncompteformation*, RNCP as well as other educational French datasets.

Publications

- **Ait-Saada M**, and Nadif M. Is Anisotropy Truely Harmful? A Case Study on Text Clustering. To appear in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, Toronto, Canada. 2023.
- **Ait-Saada M**, and Nadif M. Unsupervised Anomaly Detection in Multi-Topic Short-Text Corpora. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, Dubrovnik, Croatia, pp. 1384-1395. 2023.
- **Ait-Saada M**, and Nadif M. Contextual Word Embeddings Clustering through Multi-way Analysis: a Comparative Study. In *Advances in Intelligent Data Analysis XXI: 21st International Symposium on Intelligent Data Analysis, IDA 2023*, Louvain-la-Neuve, Belgium, Proceedings, pp. 1-14, Cham: Springer Nature Switzerland. 2023.
- **Ait-Saada M**, and Nadif M. Etude approfondie des représentations de données textuelles dans l'apprentissage non supervisé. *Revue des Nouvelles Technologies de l'Information, Extraction et Gestion des Connaissances*, RNTI-E-39:361-368. 2023.
- Boutalbi R, **Ait-Saada M**, Iurshina A, Staab S, and Nadif M. Tensor-based Graph Modularity for Text Data Clustering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*, Madrid, Spain, pp. 2227-2231. 2022.
- **Ait-Saada M**, Role F, and Nadif M. Classification non supervisée de documents à partir des modèles transformeurs. *Revue des Nouvelles Technologies de l'Information, Extraction et Gestion des Connaissances*, RNTI-E-38:331–338. 2022.
- **Ait-Saada M**, Role F, and Nadif M. How to Leverage a Multi-layered Transformer Language Model for Text Clustering: an Ensemble Approach. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21)*, Queensland, Australia, pp. 2837-2841. 2021.
- **Ait-Saada M**, Role F, and Nadif M. Unsupervised Methods for the Study of Transformer Embeddings. In *Advances in Intelligent Data Analysis XIX: 19th International Symposium on Intelligent Data Analysis, IDA 2021*, Porto, Portugal, Proceedings 19, pp. 287-300. Springer, Cham. 2021.

List of Figures

List of Tables

References

Abbe, Adeline, Grouin, Cyril, Zweigenbaum, Pierre, and Falissard, Bruno. 2016. Text mining applications in psychiatry: a systematic literature review. *International Journal of Methods in Psychiatric Research*, 25(2):86–100.

Abdi, Hervé, Williams, Lynne J., and Valentin, Dominique. 2013. Multiple factor analysis: principal component analysis for multitable and multiblock data sets. *WIREs Computational Statistics*, 5(2):149–179.

Affeldt, Séverine, Labiod, Lazhar, and Nadif, Mohamed. 2020a. Ensemble block co-clustering: A unified framework for text data. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 5–14, New York, NY, USA. Association for Computing Machinery.

Affeldt, Séverine, Labiod, Lazhar, and Nadif, Mohamed. 2021. Regularized bi-directional co-clustering. *Statistics and Computing*, 31(3):32.

Affeldt, Séverine, Labiod, Lazhar, and Nadif, Mohamed. 2020b. Spectral clustering via ensemble deep autoencoder learning (SC-EDAE). *Pattern Recognition*, 108:107522.

Affeldt, Séverine, Labiod, Lazhar, and Nadif, Mohamed. 2022. CAEclust: A consensus of autoencoders representations for clustering. *Image Processing On Line*, 12:590–603.

Agarwal, Gaurav and Kempe, David. 2008. Modularity-maximizing graph communities via mathematical programming. *The European Physical Journal B*, 66(3):409–418.

Aggarwal, Charu C. 2017. *An Introduction to Outlier Analysis*, pages 1–34. Springer International Publishing, Cham.

Ahmed, Mohiuddin, Mahmood, Abdun Naser, and Islam, Md. Rafiqul. 2016. A survey of anomaly detection techniques in financial domain. *Future Generation Computer Systems*, 55:278–288.

Ailem, Melissa, Role, François, and Nadif, Mohamed. 2015. Co-clustering document-term matrices by direct maximization of graph modularity. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, page 1807–1810, New York, NY, USA. Association for Computing Machinery.

Ailem, Melissa, Role, François, and Nadif, Mohamed. 2017a. Model-based co-clustering for the effective handling of sparse data. *Pattern Recognition*, 72:108–122.

Ailem, Melissa, Salah, Aghiles, and Nadif, Mohamed. 2017b. Non-negative matrix factorization meets word embedding. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, page 1081–1084, New York, NY, USA. Association for Computing Machinery.

Ait-Saada, Mira, Role, François, and Nadif, Mohamed. 2021. How to leverage a multi-layered Transformer language model for text clustering: An ensemble approach. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 2837–2841, New York, NY, USA. Association for Computing Machinery.

Akcay, Samet, Atapour-Abarghouei, Amir, and Breckon, Toby P. 2019. GANomaly: Semi-supervised anomaly detection via adversarial training. In *Computer Vision – ACCV 2018*, pages 622–637, Cham. Springer International Publishing.

Akoglu, Leman, Tong, Hanghang, and Koutra, Danai. 2015. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery*, 29(3):626–688.

Alexandrov, Nickolai, Tai, Shuaishuai, Wang, Wensheng, Mansueto, Locedie, Palis, Kevin, Fuentes, Roven Rommel, Ulat, Victor Jun, Chebotarov, Dmytro, Zhang, Gengyun, Li, Zhikang, Mauleon, Ramil, Hamilton, Ruaraidh Sackville, and McNally, Kenneth L. 2014. SNP-Seek database of SNPs derived from 3000 rice genomes. *Nucleic Acids Research*, 43(D1):D1023–D1027.

Alizadeh, Hosein, Minaei-Bidgoli, Behrouz, and Parvin, Hamid. 2014. Cluster ensemble selection based on a new cluster stability measure. *Intelligent Data Analysis*, 18(3):389–408.

Aljaber, Bader, Stokes, Nicola, Bailey, James, and Pei, Jian. 2010. Document clustering of scientific texts using citation contexts. *Information Retrieval*, 13(2):101–131.

Aljalbout, Elie, Golkov, Vladimir, Siddiqui, Yawar, and Cremers, Daniel. 2018. Clustering with deep learning: Taxonomy and new methods. *CoRR*, abs/1801.07648.

Allab, Kais, Labiod, Lazhar, and Nadif, Mohamed. 2016. A semi-NMF-PCA unified framework for data clustering. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):2–16.

Allab, Kais, Labiod, Lazhar, and Nadif, Mohamed. 2018. Simultaneous spectral data embedding and clustering. *IEEE transactions on neural networks and learning systems*, 29(12):6396–6401.

Allahyari, Mehdi and Kochut, Krys. 2016. Semantic tagging using topic models exploiting wikipedia category network. In *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, pages 63–70. IEEE.

Allaoui, Mebarka, Kherfi, Mohammed Lamine, and Cheriet, Abdelhakim. 2020. Considerably improving clustering algorithms using UMAP dimensionality reduction technique: A comparative study. In *Image and Signal Processing*, pages 317–325, Cham. Springer International Publishing.

Almuallim, Hussein and Dietterich, Thomas G. 1994. Learning boolean concepts in the presence of many irrelevant features. *Artificial Intelligence*, 69(1):279–305.

Alsentzer, Emily, Murphy, John, Boag, William, Weng, Wei-Hung, Jindi, Di, Naumann, Tristan, and McDermott, Matthew. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Angiulli, Fabrizio and Pizzuti, Clara. 2002. Fast outlier detection in high dimensional spaces. In *European conference on principles of data mining and knowledge discovery*, pages 15–27. Springer.

Anick, Peter G. and Vaithyanathan, Shivakumar. 1997. Exploiting clustering and phrases for context-based information retrieval. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '97, page 314–323, New York, NY, USA. Association for Computing Machinery.

Ankerst, Mihael, Breunig, Markus M., Kriegel, Hans-Peter, and Sander, Jörg. 1999. OPTICS: Ordering points to identify the clustering structure. *SIGMOD Rec.*, 28(2):49–60.

Antoun, Wissam, Baly, Fady, and Hajj, Hazem. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Argelaguet, Ricard, Velten, Britta, Arnol, Damien, Dietrich, Sascha, Zenz, Thorsten, Marioni, John C, Buettner, Florian, Huber, Wolfgang, and Stegle, Oliver. 2018. Multi-omics factor analysis—a framework for unsupervised integration of multi-omics data sets. *Molecular Systems Biology*, 14(6):e8124.

Arimond, George and Elfessi, Abdulaziz. 2001. A clustering method for categorical data in tourism market segmentation research. *Journal of Travel Research*, 39(4):391–397.

Arora, Sanjeev, Li, Yuanzhi, Liang, Yingyu, Ma, Tengyu, and Risteski, Andrej. 2016. A latent variable model approach to PMI-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399.

Arora, Sanjeev, Liang, Yingyu, and Ma, Tengyu. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *International Conference on Learning Representations*.

Aznag, Mustapha, Quafafou, Mohamed, Rochd, El Mehdi, and Jarir, Zahi. 2013. Probabilistic topic models for web services clustering and discovery. In *Service-Oriented and Cloud Computing*, pages 19–33, Berlin, Heidelberg. Springer Berlin Heidelberg.

Baghel, Rekha and Dhir, Renu. 2010. A frequent concepts based document clustering algorithm. *International Journal of Computer Applications*, 4(5):6–12.

Baker, L. Douglas and McCallum, Andrew Kachites. 1998. Distributional clustering of words for text classification. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, page 96–103, New York, NY, USA. Association for Computing Machinery.

Baldi, Pierre and Hornik, Kurt. 1989. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58.

Banerjee, Arindam, Dhillon, Inderjit S., Ghosh, Joydeep, and Sra, Suvrit. 2005. Clustering on the unit hypersphere using von Mises-Fisher distributions. *Journal of Machine Learning Research*, 6(46):1345–1382.

Banfield, Jeffrey D. and Raftery, Adrian E. 1993. Model-based gaussian and non-gaussian clustering. *Biometrics*, 49(3):803–821.

Bank, Dor, Koenigstein, Noam, and Giryes, Raja. 2020. Autoencoders. *CoRR*, abs/2003.05991.

Bassett, Danielle S, Porter, Mason A, Wymbs, Nicholas F, Grafton, Scott T, Carlson, Jean M, and Mucha, Peter J. 2013. Robust detection of dynamic community structure in networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 23(1):013142.

Becht, Etienne, Dutertre, Charles-Antoine, Kwok, Immanuel WH, Ng, Lai Guan, Ginhoux, Florent, and Newell, Evan W. 2018. Evaluation of UMAP as an alternative to t-SNE for single-cell data. *BioRxiv*, page 298430.

Belinkov, Yonatan and Glass, James. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Belkin, Mikhail and Niyogi, Partha. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

Bellman, Richard E. 1961. Adaptive control processes: a guided tour.

Beltagy, Iz, Lo, Kyle, and Cohan, Arman. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Ben-Dor, Amir, Chor, Benny, Karp, Richard, and Yakhini, Zohar. 2002. Discovering local structure in gene expression data: The order-preserving submatrix problem. In *Proceedings of the Sixth Annual International Conference on Computational Biology*, RECOMB '02, page 49–57, New York, NY, USA. Association for Computing Machinery.

Ben-Yosef, Matan and Weinshall, Daphna. 2018. Gaussian mixture generative adversarial networks for diverse datasets, and the unsupervised clustering of images. *CoRR*, abs/1808.10356.

Bengio, Yoshua, Delalleau, Olivier, Roux, Nicolas Le, Paiement, Jean-François, Vincent, Pascal, and Ouimet, Marie. 2004. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural computation*, 16(10):2197–2219.

Bengio, Yoshua, Ducharme, Réjean, and Vincent, Pascal. 2000. A neural probabilistic language model. In *Advances in Neural Information Processing Systems*, volume 13. MIT Press.

Bengio, Yoshua, Lamblin, Pascal, Popovici, Dan, and Larochelle, Hugo. 2006. Greedy layer-wise training of deep networks. 19.

Bergstra, James and Bengio, Yoshua. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(10):281–305.

Berikov, Vladimir and Pestunov, Igor. 2017. Ensemble clustering based on weighted co-association matrices: Error bound and convergence properties. *Pattern Recognition*, 63:427–436.

Berry, Michael W and Browne, Murray. 2005. Email surveillance using non-negative matrix factorization. *Computational & Mathematical Organization Theory*, 11(3):249–264.

Blondel, Vincent D, Guillaume, Jean-Loup, Lambiotte, Renaud, and Lefebvre, Etienne. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Bo, Deyu, Wang, Xiao, Shi, Chuan, Zhu, Meiqi, Lu, Emiao, and Cui, Peng. 2020. Structural deep clustering network. In *Proceedings of The Web Conference 2020*, WWW '20, page 1400–1410, New York, NY, USA. Association for Computing Machinery.

Bohlin, Jon, Skjerve, Eystein, and Ussery, David W. 2009. Analysis of genomic signatures in prokaryotes using multinomial regression and hierarchical clustering. *BMC genomics*, 10(1):1–9.

Bojanowski, Piotr, Grave, Edouard, Joulin, Armand, and Mikolov, Tomas. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Bourlard, Hervé and Kamp, Yves. 1988. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4):291–294.

Boutalbi, Rafika, Ait-Saada, Mira, Iurshina, Anastasiia, Staab, Steffen, and Nadif, Mohamed. 2022. Tensor-based graph modularity for text data clustering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2227–2231, New York, NY, USA. Association for Computing Machinery.

Boutalbi, Rafika, Labiod, Lazhar, and Nadif, Mohamed. 2021. Implicit consensus clustering from multiple graphs. *Data Mining and Knowledge Discovery*, 35:2313–2340.

Boutsidis, Christos, Zouzias, Anastasios, Mahoney, Michael W, and Drineas, Petros. 2014. Randomized dimensionality reduction for k-means clustering. *IEEE Transactions on Information Theory*, 61(2):1045–1062.

Bradley, Paul, Mangasarian, Olvi, and Street, W. 1996. Clustering via concave minimization. In *Advances in Neural Information Processing Systems*, volume 9. MIT Press.

Brbić, Maria, Zitnik, Marinka, Wang, Sheng, Pisco, Angela O, Altman, Russ B, Darmanis, Spyros, and Leskovec, Jure. 2020. MARS: discovering novel cell types across heterogeneous single-cell experiments. *Nature methods*, 17(12):1200–1206.

Breunig, Markus M, Kriegel, Hans-Peter, Ng, Raymond T, and Sander, Jörg. 2000. LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, page 93–104, New York, NY, USA. Association for Computing Machinery.

Brijs, Tom, Karlis, Dimitris, Swinnen, Gilbert, Vanhoof, Koen, Wets, Geert, and Manchanda, Puneet. 2004. A multivariate Poisson mixture model for marketing applications. *Statistica Neerlandica*, 58(3):322–348.

Buchta, Christian, Kober, Martin, Feinerer, Ingo, and Hornik, Kurt. 2012. Spherical k-means clustering. *Journal of statistical software*, 50(10):1–22.

Buechel, Sven, Sedoc, João, Schwartz, H. Andrew, and Ungar, Lyle H. 2018. Learning neural emotion analysis from 100 observations: The surprising effectiveness of pre-trained word representations. *CoRR*, abs/1810.10949.

Bullinaria, John A and Levy, Joseph P. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods*, 39(3):510–526.

Carlsson, Fredrik, Gyllensten, Amaru Cuba, Gogoulou, Evangelia, Hellqvist, Erik Ylipää, and Sahlgren, Magnus. 2021. Semantic re-tuning with contrastive tension. In *International Conference on Learning Representations*.

Carroll, J Douglas. 1968. Generalization of canonical correlation analysis to three or more sets of variables. In *Proceedings of the 76th annual convention of the American Psychological Association*, volume 3, pages 227–228. Washington, DC.

Carroll, J Douglas and Chang, Jih-Jie. 1970. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35(3):283–319.

Celeux, Gilles and Govaert, Gérard. 1995. Gaussian parsimonious clustering models. *Pattern Recognition*, 28(5):781–793.

Cer, Daniel, Yang, Yinfei, Kong, Sheng-yi, Hua, Nan, Limtiaco, Nicole, John, Rhomni St., Constant, Noah, Guajardo-Cespedes, Mario, Yuan, Steve, Tar, Chris, Sung, Yun-Hsuan, Strope, Brian, and Kurzweil, Ray. 2018. Universal sentence encoder. *CoRR*, abs/1803.11175.

Chalkidis, Ilias, Fergadiotis, Manos, Malakasiotis, Prodromos, Aletras, Nikolaos, and Androutsopoulos, Ion. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online. Association for Computational Linguistics.

Chandola, Varun, Banerjee, Arindam, and Kumar, Vipin. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3).

Chang, Wei-Chien. 1983. On using principal components before separating a mixture of two multivariate normal distributions. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 32(3):267–275.

Chang, Yen-Yu, Li, Pan, Sosic, Rok, Afifi, M. H., Schweighauser, Marco, and Leskovec, Jure. 2021. F-fade: Frequency factorization for anomaly detection in edge streams. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, page 589–597, New York, NY, USA. Association for Computing Machinery.

Charrad, Malika, Ghazzali, Nadia, Boiteau, Véronique, and Niknafs, Azam. 2014. NbClust: An R package for determining the relevant number of clusters in a data set. *Journal of Statistical Software*, 61(6):1–36.

Chavent, Marie, Lechevallier, Yves, and Briant, Olivier. 2007. Divclus-t: A monothetic divisive hierarchical clustering method. *Computational Statistics & Data Analysis*, 52(2):687–701.

Chen, Bin, Hong, Jiarong, and Wang, Yadong. 1997. The minimum feature subset selection problem. *Journal of Computer Science and Technology*, 12(2):145–153.

Chen, Jinghui, Sathe, Saket, Aggarwal, Charu, and Turaga, Deepak. 2017a. Outlier detection with autoencoder ensembles. In *Proceedings of the 2017 SIAM International Conference on Data Mining (SDM)*, pages 90–98. SIAM.

Chen, Min, Shi, Xiaobo, Zhang, Yin, Wu, Di, and Guizani, Mohsen. 2017b. Deep feature learning for medical image analysis with convolutional autoencoder neural network. *IEEE Transactions on Big Data*, 7(4):750–758.

Chen, Yan, Yin, Xiaoshi, Li, Zhoujun, Hu, Xiaohua, and Huang, Jimmy Xiangji. 2012. A LDA-based approach to promoting ranking diversity for genomics information retrieval. In *BMC genomics*, volume 13, pages 1–10. BioMed Central.

Cheng, Xingyi. 2021. Dual-view distilled BERT for sentence embedding. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 2151–2155, New York, NY, USA. Association for Computing Machinery.

Chessel, Daniel and Hanafi, M. 1996. Analyses de la co-inertie de k nuages de points. *Revue de statistique appliquée*, 44(2):35–60.

Chung, Junyoung, Gulcehre, Caglar, Cho, Kyunghyun, and Bengio, Yoshua. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.

Clark, Kevin, Khandelwal, Urvashi, Levy, Omer, and Manning, Christopher D. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Clark, Kevin, Luong, Minh-Thang, Le, Quoc V., and Manning, Christopher D. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Clauset, Aaron, Newman, Mark EJ, and Moore, Cristopher. 2004. Finding community structure in very large networks. *Physical review E*, 70(6):066111.

Conneau, Alexis, Khandelwal, Kartikay, Goyal, Naman, Chaudhary, Vishrav, Wenzek, Guillaume, Guzmán, Francisco, Grave, Edouard, Ott, Myle, Zettlemoyer, Luke, and Stoyanov, Veselin. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Conneau, Alexis, Kiela, Douwe, Schwenk, Holger, Barrault, Loïc, and Bordes, Antoine. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Cortal, Gustave. 2022. Covid-19 - french news dataset.

Cunningham, John P. and Ghahramani, Zoubin. 2015. Linear dimensionality reduction: Survey, insights, and generalizations. *Journal of Machine Learning Research*, 16(89):2859–2900.

Dai, Andrew M and Le, Quoc V. 2015. Semi-supervised sequence learning. *Advances in neural information processing systems*, 28.

Das, Arjun, Ganguly, Debasis, and Garain, Utpal. 2017. Named entity recognition with word embeddings and wikipedia categories for a low-resource language. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 16(3).

Das, Mahashweta and Parthasarathy, Srinivasan. 2009. Anomaly detection and spatio-temporal analysis of global climate system. In *Proceedings of the third international workshop on knowledge discovery from sensor data*, pages 142–150.

Davies, David L and Bouldin, Donald W. 1979. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227.

De Lathauwer, Lieven, De Moor, Bart, and Vandewalle, Joos. 2000. On the best rank-1 and rank-($R_1, R_2, \ldots, R_N$) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1324–1342.

De Soete, Geert and Carroll, J. Douglas. 1994. K-means clustering in a low-dimensional Euclidean space. In *New Approaches in Classification and Data Analysis*, pages 212–219, Berlin, Heidelberg. Springer Berlin Heidelberg.

de Vries, Wietse, van Cranenburgh, Andreas, and Nissim, Malvina. 2020. What's so special about BERT's layers? a closer look at the NLP pipeline in monolingual and multilingual models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4339–4350, Online. Association for Computational Linguistics.

Decherchi, Sergio, Tacconi, Simone, Redi, Judith, Leoncini, Alessio, Sangiacomo, Fabio, and Zunino, Rodolfo. 2009. Text clustering for digital forensics analysis. In *Computational Intelligence in Security for Information Systems*, pages 29–36. Springer.

Deerwester, Scott, Dumais, Susan T, Furnas, George W, Landauer, Thomas K, and Harshman, Richard. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Dempster, A. P., Laird, N. M., and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Demšar, Janez. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1):1–30.

Deng, Qingshan and Mei, Guoping. 2009. Combining self-organizing map and k-means clustering for detecting fraudulent financial statements. In *2009 IEEE International Conference on Granular Computing*, pages 126–131.

Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, and Toutanova, Kristina. 2019. BERT: Pre-training of deep bidirectional Transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dhillon, Inderjit S. 2001. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, page 269–274, New York, NY, USA. Association for Computing Machinery.

Dhillon, Inderjit S., Mallela, Subramanyam, and Modha, Dharmendra S. 2003. Information-theoretic co-clustering. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, page 89–98, New York, NY, USA. Association for Computing Machinery.

Dhillon, Inderjit S and Modha, Dharmendra S. 2001. Concept decompositions for large sparse text data using clustering. *Machine learning*, 42(1):143–175.

Dietterich, Thomas G. 2000. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer.

Dilokthanakul, Nat, Mediano, Pedro A. M., Garnelo, Marta, Lee, Matthew C. H., Salimbeni, Hugh, Arulkumaran, Kai, and Shanahan, Murray. 2016. Deep unsupervised clustering with Gaussian mixture variational autoencoders. *CoRR*, abs/1611.02648.

Dimitriadou, Evgenia, Weingessel, Andreas, and Hornik, Kurt. 2002. A combination scheme for fuzzy clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, 16(07):901–912.

Ding, Xueying, Zhao, Lingxiao, and Akoglu, Leman. 2022a. Hyperparameter sensitivity in deep outlier detection: Analysis and a scalable hyper-ensemble solution. In *Advances in Neural Information Processing Systems*.

Ding, Yue, Martinkus, Karolis, Pascual, Damian, Clematide, Simon, and Wattenhofer, Roger. 2022b. On isotropy calibration of Transformer models. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Drineas, Petros, Frieze, Alan, Kannan, Ravi, Vempala, Santosh, and Vinay, V. 2004. Clustering large graphs via the singular value decomposition. *Machine learning*, 56(1):9–33.

Duch, Jordi and Arenas, Alex. 2005. Community detection in complex networks using extremal optimization. *Physical review E*, 72(2):027104.

Dumais, Susan T, Furnas, George W, Landauer, Thomas K, Deerwester, Scott, and Harshman, Richard. 1988. Using latent semantic analysis to improve access to textual information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 281–285.

Dunn, Joseph C. 1974. Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, 4(1):95–104.

Dy, Jennifer G and Brodley, Carla E. 2004. Feature selection for unsupervised learning. *Journal of machine learning research*, 5(Aug):845–889.

Edwards, Anthony WF and Cavalli-Sforza, Luigi Luca. 1965. A method for cluster analysis. *Biometrics*, pages 362–375.

Eren, Maksim E., Moore, Juston S., Skau, Erik, Moore, Elisabeth, Bhattarai, Manish, Chennupati, Gopinath, and Alexandrov, Boian S. 2022. General-purpose unsupervised cyber anomaly detection via non-negative tensor factorization. *Digital Threats*. Just Accepted.

Escofier, B et al. 1983. Méthode pour l'analyse de plusieurs groupes de variables. application à la caractérisation de vins rouges du Val de Loire. *Revue de statistique appliquée*, 31(2):43–59.

Escoufier, Yves. 1980. L'analyse conjointe de plusieurs matrices de données. *Biométrie et temps*, 58:59–76.

Eskin, Eleazar, Arnold, Andrew, Prerau, Michael, Portnoy, Leonid, and Stolfo, Sal. 2002. A geometric framework for unsupervised anomaly detection. In *Applications of data mining in computer security*, pages 77–101. Springer.

Espadoto, Mateus, Martins, Rafael M., Kerren, Andreas, Hirata, Nina S. T., and Telea, Alexandru C. 2021. Toward a quantitative survey of dimension reduction techniques. *IEEE Transactions on Visualization and Computer Graphics*, 27(3):2153–2173.

Ester, Martin, Kriegel, Hans-Peter, Sander, Jörg, Xu, Xiaowei, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Knowledge Discovery and Data Mining*, volume 96, pages 226–231.

Ethayarajh, Kawin. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Fabius, Otto, van Amersfoort, Joost R., and Kingma, Diederik P. 2015. Variational recurrent auto-encoders. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*.

Fan, Wentao, Bouguila, Nizar, and Ziou, Djemel. 2011. Unsupervised anomaly intrusion detection via localized bayesian feature selection. In *2011 IEEE 11th International Conference on Data Mining*, pages 1032–1037. IEEE.

Fard, Maziar Moradi, Thonet, Thibaut, and Gaussier, Eric. 2020. Deep k-means: Jointly clustering with k-means and learning representations. *Pattern Recognition Letters*, 138:185–192.

Fauconnier, Jean-Philippe. 2015. French word embeddings.

Febrissy, Mickael and Nadif, Mohamed. 2020. A consensus approach to improve NMF document clustering. In *Advances in Intelligent Data Analysis XVIII: 18th International Symposium on Intelligent Data Analysis, IDA 2020, Konstanz, Germany, April 27–29, 2020, Proceedings 18*, pages 171–183. Springer.

Febrissy, Mickael, Salah, Aghiles, Ailem, Melissa, and Nadif, Mohamed. 2022. Improving NMF clustering by leveraging contextual relationships among words. *Neurocomputing*, 495:105–117.

Fern, Xiaoli Zhang and Brodley, Carla E. 2004. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the twenty-first international conference on Machine learning*, page 36.

Fettal, Chakib, Labiod, Lazhar, and Nadif, Mohamed. 2023. Simultaneous linear multi-view attributed graph representation learning and clustering. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23)*.

Fodor, Imola K. 2002. A survey of dimension reduction techniques. Technical report, Lawrence Livermore National Lab., CA (US).

France, Stephen L and Akkucuk, Ulas. 2021. A review, framework, and R toolkit for exploring, evaluating, and comparing visualization methods. *The Visual Computer*, 37(3):457–475.

Fred, Ana. 2001. Finding consistent clusters in data partitions. In *International Workshop on Multiple Classifier Systems*, pages 309–318. Springer.

Gao, Byron J., Griffith, Obi L., Ester, Martin, and Jones, Steven J. M. 2006. Discovering significant OPSM subspace clusters in massive gene expression data. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 922–928, New York, NY, USA. Association for Computing Machinery.

Gao, Jun, He, Di, Tan, Xu, Qin, Tao, Wang, Liwei, and Liu, Tieyan. 2019. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*.

Gao, Tianyu, Yao, Xingcheng, and Chen, Danqi. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Garten, Justin, Sagae, Kenji, Ustun, Volkan, and Dehghani, Morteza. 2015. Combining distributed vector representations for words. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 95–101, Denver, Colorado. Association for Computational Linguistics.

Ghahramani, Zoubin, Hinton, Geoffrey E, et al. 1996. The EM algorithm for mixtures of factor analyzers. Technical report, CRG-TR-96-1, University of Toronto.

Ghojogh, Benyamin, Ghodsi, Ali, Karray, Fakhri, and Crowley, Mark. 2021. Factor analysis, probabilistic principal component analysis, variational inference, and variational autoencoder: Tutorial and survey. *arXiv preprint arXiv:2101.00734*.

Gogoi, Prasanta, Bhattacharyya, Dhruba K, Borah, Bhogeswar, and Kalita, Jugal K. 2011. A survey of outlier detection methods in network anomaly identification. *The Computer Journal*, 54(4):570–588.

Goldberg, Yoav. 2019. Assessing BERT's syntactic abilities. *CoRR*, abs/1901.05287.

Goodfellow, Ian, Bengio, Yoshua, and Courville, Aaron. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Gopal, Siddharth and Yang, Yiming. 2014. Von Mises-Fisher clustering models. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 154–162, Bejing, China. PMLR.

Govaert, G. 2009. Data analysis. ISTE Ltd and John Wiley & Sons. *Inc.(Ed.), 318p.*

Govaert, Gérard and Nadif, Mohamed. 2013. *Co-clustering: models, algorithms and applications*. John Wiley & Sons.

Grave, Edouard, Bojanowski, Piotr, Gupta, Prakhar, Joulin, Armand, and Mikolov, Tomas. 2018. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Greene, Derek and Cunningham, Pádraig. 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 377–384, New York, NY, USA. Association for Computing Machinery.

Guha, Sudipto, Rastogi, Rajeev, and Shim, Kyuseok. 1998. CURE: An efficient clustering algorithm for large databases. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, SIGMOD '98, page 73–84, New York, NY, USA. Association for Computing Machinery.

Guo, Xifeng, Gao, Long, Liu, Xinwang, and Yin, Jianping. 2017. Improved deep embedded clustering with local structure preservation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 1753–1759.

Gupta, Ashim, Kvernadze, Giorgi, and Srikumar, Vivek. 2021. BERT & family eat word salad: Experiments with text understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12946–12954.

Hadifar, Amir, Sterckx, Lucas, Demeester, Thomas, and Develder, Chris. 2019. A self-training approach for short text clustering. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 194–199, Florence, Italy. Association for Computational Linguistics.

Haibe-Kains, Benjamin, Adam, George Alexandru, Hosny, Ahmed, Khodakarami, Farnoosh, Waldron, Levi, Wang, Bo, McIntosh, Chris, Goldenberg, Anna, Kundaje, Anshul, Greene, Casey S, et al. 2020. Transparency and reproducibility in artificial intelligence. *Nature*, 586(7829):E14–E16.

Halpern, Yoni, Horng, Steven, Nathanson, Larry A., Shapiro, Nathan I., and Sontag, David. 2012. A comparison of dimensionality reduction techniques for unstructured clinical text. In *ICML 2012 Workshop on Clinical Data Analysis*.

Han, Changhee, Rundo, Leonardo, Murao, Kohei, Noguchi, Tomoyuki, Shimahara, Yuki, Milacski, Zoltán Ádám, Koshino, Saori, Sala, Evis, Nakayama, Hideki, and Satoh, Shin'ichi. 2021. MADGAN: Unsupervised medical anomaly detection GAN using multiple adjacent brain MRI slice reconstruction. *BMC bioinformatics*, 22(2):1–20.

Hanafi, Mohamed, Kohler, Achim, and Qannari, El-Mostafa. 2011. Connections between multiple co-inertia analysis and consensus principal component analysis. *Chemometrics and intelligent laboratory systems*, 106(1):37–40.

Hao, Liyang, Liang, Siqi, Ye, Jinmian, and Xu, Zenglin. 2018. TensorD: A tensor decomposition library in Tensorflow. *Neurocomputing*, 318:196–200.

Hao, Yaru, Dong, Li, Wei, Furu, and Xu, Ke. 2019. Visualizing and Understanding the Effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4141–4150, Hong Kong, China. Association for Computational Linguistics.

Harris, Zellig S. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Harshman, Richard A et al. 1970. Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multimodal factor analysis.

Hawkins, Douglas M. 1980. *Identification of outliers*, volume 11. Springer.

He, Xiaofei, Cai, Deng, Liu, Haifeng, and Ma, Wei-Ying. 2004. Locality preserving indexing for document representation. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, page 96–103, New York, NY, USA. Association for Computing Machinery.

He, Xiaofei and Niyogi, Partha. 2003. Locality preserving projections. *Advances in neural information processing systems*, 16.

Heard, Nicholas A, Weston, David J, Platanioti, Kiriaki, and Hand, David J. 2010. Bayesian anomaly detection methods for social networks. *The Annals of Applied Statistics*, 4(2):645–662.

Hempstalk, Kathryn, Frank, Eibe, and Witten, Ian H. 2008. One-class classification by combining density and class probability estimation. In *Machine Learning and Knowledge Discovery in Databases*, pages 505–519, Berlin, Heidelberg. Springer Berlin Heidelberg.

Hinneburg, Alexander and Keim, Daniel A. 1998. *An efficient approach to clustering in large multimedia databases with noise*, volume 98. Bibliothek der Universität Konstanz.

Hinton, Geoffrey E and Roweis, Sam. 2002. Stochastic neighbor embedding. *Advances in neural information processing systems*, 15.

Hinton, Geoffrey E and Salakhutdinov, Ruslan R. 2006. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507.

Hodge, Victoria and Austin, Jim. 2004. A survey of outlier detection methodologies. *Artificial intelligence review*, 22(2):85–126.

Hooi, Bryan, Song, Hyun Ah, Beutel, Alex, Shah, Neil, Shin, Kijung, and Faloutsos, Christos. 2016. FRAUDAR: Bounding graph fraud in the face of camouflage. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 895–904, New York, NY, USA. Association for Computing Machinery.

Hotho, Andreas, Maedche, Alexander, and Staab, Steffen. 2001. Ontology-based text clustering. In *International Joint Conference on Artificial Intelligence*. Workshop "Text Learning: Beyond Supervision".

Hsieh, Ruei-Jie, Chou, Jerry, and Ho, Chih-Hsiang. 2019. Unsupervised online anomaly detection on multivariate sensing time series data for smart manufacturing. In *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*, pages 90–97.

Huang, Hao and Kasiviswanathan, Shiva Prasad. 2015. Streaming anomaly detection using randomized matrix sketching. *Proceedings of the VLDB Endowment*, 9(3):192–203.

Huang, Junjie, Tang, Duyu, Zhong, Wanjun, Lu, Shuai, Shou, Linjun, Gong, Ming, Jiang, Daxin, and Duan, Nan. 2021. WhiteningBERT: An easy unsupervised sentence embedding approach. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 238–244, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Huang, Peihao, Huang, Yan, Wang, Wei, and Wang, Liang. 2014. Deep embedding network for clustering. In *2014 22nd International conference on pattern recognition*, pages 1532–1537. IEEE.

Hubert, Lawrence and Arabie, Phipps. 1985. Comparing partitions. *Journal of classification*, 2(1):193–218.

Hutter, Frank, Kotthoff, Lars, and Vanschoren, Joaquin. 2019. *Automated machine learning: methods, systems, challenges*. Springer Nature.

Hwang, Heungsun, Dillon, William R, and Takane, Yoshio. 2006. An extension of multiple correspondence analysis for identifying heterogeneous subgroups of respondents. *Psychometrika*, 71(1):161–171.

Ilin, Alexander and Raiko, Tapani. 2010. Practical approaches to principal component analysis in the presence of missing values. *Journal of Machine Learning Research*, 11(66):1957–2000.

Jablonski, James A, Bihl, Trevor J, and Bauer, Kenneth W. 2015. Principal component reconstruction error for hyperspectral anomaly detection. *IEEE Geoscience and Remote Sensing Letters*, 12(8):1725–1729.

Jafarzadegan, Mohammad, Safi-Esfahani, Faramarz, and Beheshti, Zahra. 2019. Combining hierarchical clustering approaches using the PCA method. *Expert Systems with Applications*, 137:1–10.

Jawahar, Ganesh, Sagot, Benoît, and Seddah, Djamé. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Ji, Pan, Zhang, Tong, Li, Hongdong, Salzmann, Mathieu, and Reid, Ian. Deep subspace clustering networks. 30.

Jia, Jianhua, Xiao, Xuan, Liu, Bingxiang, and Jiao, Licheng. 2011. Bagging-based spectral clustering ensemble selection. *Pattern Recognition Letters*, 32(10):1456–1467.

Jia, Kui, Sun, Lin, Gao, Shenghua, Song, Zhan, and Shi, Bertram E. 2015. Laplacian auto-encoders: An explicit learning of nonlinear data manifold. *Neurocomputing*, 160:250–260.

Jia, Yuting, Zhang, Qinqin, Zhang, Weinan, and Wang, Xinbing. 2019. CommunityGAN: Community detection with generative adversarial nets. In *The World Wide Web Conference*, WWW '19, page 784–794, New York, NY, USA. Association for Computing Machinery.

Jiang, Mon-Fong, Tseng, Shian-Shyong, and Su, Chih-Ming. 2001. Two-phase clustering process for outliers detection. *Pattern recognition letters*, 22(6-7):691–700.

Jollois, François-Xavier and Nadif, Mohamed. 2002. Clustering large categorical data. In *Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, PAKDD '02, page 257–263, Berlin, Heidelberg. Springer-Verlag.

Joudaki, Hossein, Rashidian, Arash, Minaei-Bidgoli, Behrouz, Mahmoodi, Mahmood, Geraili, Bijan, Nasiri, Mahdi, and Arab, Mohammad. 2015. Using data mining to detect health care fraud and abuse: a review of literature. *Global journal of health science*, 7(1):194–202.

Kadhim, Ammar Ismael, Cheah, Yu-N, and Ahamed, Nurul Hashimah. 2014. Text document preprocessing and dimension reduction techniques for text document clustering. In *4th International Conference on Artificial Intelligence with Applications in Engineering and Technology*, pages 69–73.

Kakkonen, Tuomo, Myller, Niko, Sutinen, Erkki, and Timonen, Jari. 2008. Comparison of dimension reduction methods for automated essay grading. *Journal of Educational Technology & Society*, 11(3):275–288.

Kalyan, Katikapalli Subramanyam, Rajasekharan, Ajit, and Sangeetha, Sivanesan. 2021. AMMUS : A survey of Transformer-based pretrained models in natural language processing. *CoRR*, abs/2108.05542.

Kanehara, Hideaki, Murakami, Yuma, Shimamura, Jumpei, Takahashi, Takeshi, Inoue, Daisuke, and Murata, Noboru. 2019. Real-time botnet detection using nonnegative Tucker decomposition. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, SAC '19, page 1337–1344, New York, NY, USA. Association for Computing Machinery.

Kannan, Ramakrishnan, Woo, Hyenkyun, Aggarwal, Charu C., and Park, Haesun. 2017. *Outlier Detection for Text Data*, pages 489–497.

Karami, Amir. 2019. Application of fuzzy clustering for text data dimensionality reduction. *International Journal of Knowledge Engineering and Data Mining*, 6(3):289–306.

Karlis, Dimitris and Meligkotsidou, Loukia. 2007. Finite mixtures of multivariate Poisson distributions with application. *Journal of statistical Planning and Inference*, 137(6):1942–1960.

Kart, Turkay, Bai, Wenjia, Glocker, Ben, and Rueckert, Daniel. 2021. DeepMCAT: Large-scale deep clustering for medical image categorization. In *Deep Generative Models, and Data Augmentation, Labelling, and Imperfections*, pages 259–267. Springer.

Kaufman, Leonard and Rousseeuw, Peter J. 2009. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons.

Keller, Fabian, Muller, Emmanuel, and Bohm, Klemens. 2012. Hics: High contrast subspaces for density-based outlier ranking. In *2012 IEEE 28th international conference on data engineering*, pages 1037–1048. IEEE.

Kenter, Tom, Borisov, Alexey, and de Rijke, Maarten. 2016. Siamese CBOW: Optimizing word embeddings for sentence representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 941–951, Berlin, Germany. Association for Computational Linguistics.

Khan, Shawez, Taverna, Federico, Rohlenova, Katerina, Treps, Lucas, Geldhof, Vincent, de Rooij, Laura, Sokol, Liliana, Pircher, Andreas, Conradi, Lena-Christin, Kalucka, Joanna, Schoonjans, Luc, Eelen, Guy, Dewerchin, Mieke, Karakach, Tobias, Li, Xuri, Goveia, Jermaine, and Carmeliet, Peter. 2018. EndoDB: a database of endothelial cell transcriptomics data. *Nucleic Acids Research*, 47(D1):D736–D744.

Kim, Junhong, Park, Minsik, Kim, Haedong, Cho, Suhyoun, and Kang, Pilsung. 2019. Insider threat detection based on user behavior modeling and anomaly detection algorithms. *Applied Sciences*, 9(19):4018.

Kim, Yoon. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Kingma, Diederik P and Welling, Max. 2014. Auto-encoding variational Bayes. In *International Conference on Learning Representations*.

Kipf, Thomas N. and Welling, Max. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Kiros, Ryan, Zhu, Yukun, Salakhutdinov, Russ R, Zemel, Richard, Urtasun, Raquel, Torralba, Antonio, and Fidler, Sanja. 2015. Skip-thought vectors. *Advances in neural information processing systems*, 28.

Knorr, Edwin M. and Ng, Raymond T. 1998. Algorithms for mining distance-based outliers in large datasets. In *Proceedings of the 24th International Conference on Very Large Data Bases*, VLDB '98, page 392–403, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Kok, Stanley and Domingos, Pedro. 2008. Extracting semantic networks from text via relational clustering. In *Machine Learning and Knowledge Discovery in Databases*, pages 624–639, Berlin, Heidelberg. Springer Berlin Heidelberg.

Kolda, Tamara G. and Bader, Brett W. 2009. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500.

Kolda, Tamara G, Bader, Brett W, and Kenny, Joseph P. 2005. Higher-order web link analysis using multilinear algebra. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 8–pp. IEEE.

Koller, Daphne and Sahami, Mehran. 1996. Toward optimal feature selection. Technical Report 1996-77, Stanford InfoLab. Previous number = SIDL-WP-1996-0032.

Kossaifi, Jean, Panagakis, Yannis, Anandkumar, Anima, and Pantic, Maja. 2019. Tensorly: Tensor learning in Python. *Journal of Machine Learning Research*, 20(26):1–6.

Kovaleva, Olga, Romanov, Alexey, Rogers, Anna, and Rumshisky, Anna. 2019. Revealing the Dark Secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4364–4373, Hong Kong, China. Association for Computational Linguistics.

Kruskal, Joseph B and Wish, Myron. 1978. Multidimensional scaling. *Paper Series on Quantitative Applications in the Social Sciences*.

Kudo, Taku and Richardson, John. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Kumar, Naveen, Venugopal, Deepak, Qiu, Liangfei, and Kumar, Subodha. 2019. Detecting anomalous online reviewers: An unsupervised approach using mixture models. *Journal of Management Information Systems*, 36(4):1313–1346.

Kuna, H.D., García-Martinez, R., and Villatoro, F.R. 2014. Outlier detection in audit logs for application systems. *Information Systems*, 44:22–33.

Kwak, Nojun and Choi, Chong-Ho. 2002. Input feature selection for classification problems. *IEEE transactions on neural networks*, 13(1):143–159.

Labiod, Lazhar and Nadif, Mohamed. 2011. Co-clustering for binary and categorical data with maximum modularity. In *2011 IEEE 11th International Conference on Data Mining*, pages 1140–1145.

Labiod, Lazhar and Nadif, Mohamed. 2021. Efficient regularized spectral data embedding. *Advances in Data Analysis and Classification*, 15(1):99–119.

Lamirel, Jean-Charles, Dugué, Nicolas, and Cuxac, Pascal. 2016. New efficient clustering quality indexes. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3649–3657.

Lan, Zhenzhong, Chen, Mingda, Goodman, Sebastian, Gimpel, Kevin, Sharma, Piyush, and Soricut, Radu. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Lau, Jey Han and Baldwin, Timothy. 2016. An empirical evaluation of doc2vec with practical insights into document embedding generation. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 78–86, Berlin, Germany. Association for Computational Linguistics.

Le, Hang, Vial, Loïc, Frej, Jibril, Segonne, Vincent, Coavoux, Maximin, Lecouteux, Benjamin, Allauzen, Alexandre, Crabbé, Benoit, Besacier, Laurent, and Schwab, Didier. 2020. FlauBERT: Unsupervised language model pre-training for French. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2479–2490, Marseille, France. European Language Resources Association.

Le, Quoc and Mikolov, Tomas. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196, Bejing, China. PMLR.

Lê, Sébastien, Josse, Julie, and Husson, François. 2008. FactoMineR: A package for multivariate analysis. *Journal of Statistical Software*, 25(1):1–18.

Lebret, Rémi and Collobert, Ronan. 2014. Word embeddings through hellinger PCA. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 482–490, Gothenburg, Sweden. Association for Computational Linguistics.

LeCun, Yann, Bengio, Yoshua, et al. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995.

Lee, Daniel D and Seung, H Sebastian. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791.

Lee, Jinhyuk, Yoon, Wonjin, Kim, Sungdong, Kim, Donghyeon, Kim, Sunkyu, So, Chan Ho, and Kang, Jaewoo. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Lehmann, Jens, Isele, Robert, Jakob, Max, Jentzsch, Anja, Kontokostas, Dimitris, Mendes, Pablo N, Hellmann, Sebastian, Morsey, Mohamed, Van Kleef, Patrick, Auer, Sören, et al. 2015. DBpedia – a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Leighton, Tom and Rao, Satish. 1989. An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. Technical report, Massachusetts Inst Of Tech Cambridge Microsystems Research Center.

Levy, Omer and Goldberg, Yoav. 2014. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, 27.

Levy, Omer, Goldberg, Yoav, and Dagan, Ido. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Lewis, Mike, Liu, Yinhan, Goyal, Naman, Ghazvininejad, Marjan, Mohamed, Abdelrahman, Levy, Omer, Stoyanov, Veselin, and Zettlemoyer, Luke. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Li, Baichuan and King, Irwin. 2010. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, page 1585–1588, New York, NY, USA. Association for Computing Machinery.

Li, Bohan, Zhou, Hao, He, Junxian, Wang, Mingxuan, Yang, Yiming, and Li, Lei. 2020a. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130, Online. Association for Computational Linguistics.

Li, Wenyuan, Zhang, Shihua, Liu, Chun-Chi, and Zhou, Xianghong Jasmine. 2012. Identifying multi-layer gene regulatory modules from multi-dimensional genomic data. *Bioinformatics*, 28(19):2458–2466.

Li, Yutong, Cai, Juanjuan, and Wang, Jingling. 2020b. A text document clustering method based on weighted BERT model. In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 1, pages 1426–1430.

Lim, Kart-Leong, Jiang, Xudong, and Yi, Chenyu. 2020. Deep clustering with variational autoencoder. *IEEE Signal Processing Letters*, 27:231–235.

Lin, Qingjian, Yin, Ruiqing, Li, Ming, Bredin, Hervé, and Barras, Claude. 2019. LSTM based similarity measurement with spectral clustering for speaker diarization. *arXiv preprint arXiv:1907.10393*.

Liu, Fangyu, Vulić, Ivan, Korhonen, Anna, and Collier, Nigel. 2021. Fast, effective, and self-supervised: Transforming masked language models into universal lexical and sentence encoders. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1442–1459, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Liu, Fei Tony, Ting, Kai Ming, and Zhou, Zhi-Hua. 2008. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422.

Liu, Huan and Setiono, Rudy. 1996. A probabilistic approach to feature selection - a filter solution. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, ICML'96, page 319–327, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Liu, Nelson F., Gardner, Matt, Belinkov, Yonatan, Peters, Matthew E., and Smith, Noah A. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Liu, Ning, Zhang, Benyu, Yan, Jun, Chen, Zheng, Liu, Wenyin, Bai, Fengshan, and Chien, Leefeng. 2005. Text representation: from vector to tensor. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 4 pp.–.

Liu, Yinhan, Ott, Myle, Goyal, Naman, Du, Jingfei, Joshi, Mandar, Chen, Danqi, Levy, Omer, Lewis, Mike, Zettlemoyer, Luke, and Stoyanov, Veselin. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Llobell, Fabien, Cariou, Véronique, Vigneau, Evelyne, Labenne, Amaury, and Qannari, El Mostafa. 2020. Analysis and clustering of multiblock datasets by means of the STATIS and CLUSTATIS methods. application to sensometrics. *Food Quality and Preference*, 79:103520.

Lloyd, Stuart. 1982. Least squares quantization in PCM. *IEEE transactions on information theory*, 28(2):129–137.

Lorenzo, Pablo Ribalta, Nalepa, Jakub, Kawulok, Michal, Ramos, Luciano Sanchez, and Pastor, José Ranilla. 2017. Particle swarm optimization for hyper-parameter selection in deep neural networks. In *Proceedings of the genetic and evolutionary computation conference*, pages 481–488.

Lu, Jinghui and MacNamee, Brian. 2020. Investigating the effectiveness of representations based on pretrained Transformer-based language models in active learning for labelling text datasets. *CoRR*, abs/2004.13138.

Lund, Kevin and Burgess, Curt. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers*, 28(2):203–208.

MacQueen, James et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Mahadevan, Vijay, Li, Weixin, Bhalodia, Viral, and Vasconcelos, Nuno. 2010. Anomaly detection in crowded scenes. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1975–1981.

Manevitz, Larry and Yousef, Malik. 2007. One-class document classification via neural networks. *Neurocomputing*, 70(7):1466–1481. Advances in Computational Intelligence and Learning.

Manevitz, Larry M and Yousef, Malik. 2001. One-class SVMs for document classification. *Journal of machine Learning research*, 2(Dec):139–154.

Manolache, Andrei, Brad, Florin, and Burceanu, Elena. 2021. DATE: Detecting anomalies in text via self-supervision of Transformers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 267–277, Online. Association for Computational Linguistics.

Marcos Alvarez, Alejandro, Yamada, Makoto, Kimura, Akisato, and Iwata, Tomoharu. 2013. Clustering-based anomaly detection in multi-view data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, CIKM '13, page 1545–1548, New York, NY, USA. Association for Computing Machinery.

Markos, Angelos, D'Enza, Alfonso Iodice, and van de Velden, Michel. 2019. Beyond tandem analysis: Joint dimension reduction and clustering in R. *Journal of Statistical Software*, 91:1–24.

Markos, Angelos, Moschidis, Odysseas, and Chadjipantelis, Theodore. 2020. Sequential dimension reduction and clustering of mixed-type data. *International Journal of Data Analysis Techniques and Strategies*, 12(3):228–246.

Martin, Louis, Muller, Benjamin, Ortiz Suárez, Pedro Javier, Dupont, Yoann, Romary, Laurent, de la Clergerie, Éric, Seddah, Djamé, and Sagot, Benoît. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

McCallum, Andrew, Nigam, Kamal, et al. 1998. A comparison of event models for naive Bayes text classification. In *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, volume 752, pages 41–48. Madison, WI.

McCann, Bryan, Bradbury, James, Xiong, Caiming, and Socher, Richard. 2017. Learned in translation: Contextualized word vectors. *Advances in neural information processing systems*, 30.

McConville, Ryan, Santos-Rodriguez, Raul, Piechocki, Robert J, and Craddock, Ian. 2021. N2d:(not too) deep clustering via clustering the local manifold of an autoencoded embedding. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5145–5152. IEEE.

McInnes, Leland, Healy, John, and Melville, James. 2018. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

McLachlan, Geoffrey J. and Peel, David. 2004. *Finite mixture models*. John Wiley & Sons.

McLachlan, Geoffrey J, Peel, David, and Bean, Richard W. 2003. Modelling high-dimensional data by mixtures of factor analyzers. *Computational Statistics & Data Analysis*, 41(3-4):379–388.

Meng, Chen. 2019. *mogsa: Multiple omics data integrative clustering and gene set analysis*. R package version 1.20.0.

Meng, Chen, Zeleznik, Oana A, Thallinger, Gerhard G, Kuster, Bernhard, Gholami, Amin M, and Culhane, Aedín C. 2016. Dimension reduction techniques for the integrative analysis of multi-omics data. *Briefings in bioinformatics*, 17(4):628–641.

Meng, Yu, Huang, Jiaxin, Wang, Guangyuan, Zhang, Chao, Zhuang, Honglei, Kaplan, Lance, and Han, Jiawei. 2019. Spherical text embedding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26.

Mikolov, Tomas, Yih, Wen-tau, and Zweig, Geoffrey. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Milligan, Glenn W and Cooper, Martha C. 1987. Methodology review: Clustering methods. *Applied psychological measurement*, 11(4):329–354.

Mittal, Himanshu, Pandey, Avinash Chandra, Pal, Raju, and Tripathi, Ashish. 2021. A new clustering method for the diagnosis of CoVID19 using medical images. *Applied Intelligence*, 51(5):2988–3011.

Morin, Frederic and Bengio, Yoshua. 2005. Hierarchical probabilistic neural network language model. In *International workshop on artificial intelligence and statistics*, pages 246–252. PMLR.

Moriya, Takayasu, Roth, Holger R, Nakamura, Shota, Oda, Hirohisa, Nagara, Kai, Oda, Masahiro, and Mori, Kensaku. 2018. Unsupervised segmentation of 3D medical images based on clustering and deep representation learning. In *Medical Imaging 2018: Biomedical Applications in Molecular, Structural, and Functional Imaging*, volume 10578, pages 483–489. SPIE.

Mu, Jiaqi and Viswanath, Pramod. 2018. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*.

Mukherjee, Subhabrata and Hassan Awadallah, Ahmed. 2020. XtremeDistil: Multi-stage distillation for massive multilingual models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2221–2234, Online. Association for Computational Linguistics.

Müller, Emmanuel, Assent, Ira, Iglesias, Patricia, Mülle, Yvonne, and Böhm, Klemens. 2012. Outlier ranking via subspace analysis in multiple views of the data. In *2012 IEEE 12th international conference on data mining*, pages 529–538. IEEE.

Munir, Mohsin, Siddiqui, Shoaib Ahmed, Dengel, Andreas, and Ahmed, Sheraz. 2018. DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access*, 7:1991–2005.

Nadif, M and Govaert, G. 1997. Clustering for binary data and mixture models—choice of the model. *Applied Stochastic Models in Business and Industry*, 13(3-4):269–278.

Nadif, Mohamed and Role, François. 2021. Unsupervised and self-supervised deep learning approaches for biomedical text mining. *Briefings in Bioinformatics*, 22(2):1592–1603.

Nemenyi, Peter Bjorn. 1963. *Distribution-free multiple comparisons.* Princeton University.

Newman, Mark EJ. 2004a. Analysis of weighted networks. *Physical review E*, 70(5):056131.

Newman, Mark EJ. 2004b. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133.

Newman, Mark EJ. 2006. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582.

Newman, Mark EJ and Girvan, Michelle. 2004. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113.

Ng, Andrew, Jordan, Michael, and Weiss, Yair. 2001. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14.

Nicosia, Vincenzo, Mangioni, Giuseppe, Carchiolo, Vincenza, and Malgeri, Michele. 2009. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(03):P03024.

Niu, Cheng, Li, Wei, Srihari, Rohini K., Li, Huifeng, and Crist, Laurie. 2004. Context clustering for word sense disambiguation based on modeling pairwise context similarities. In *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 187–190, Barcelona, Spain. Association for Computational Linguistics.

Niu, Donglin, Dy, Jennifer, and Jordan, Michael I. 2011. Dimensionality reduction for spectral clustering. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 552–560, Fort Lauderdale, FL, USA. PMLR.

Oza, Poojan and Patel, Vishal M. 2018. One-class convolutional neural network. *IEEE Signal Processing Letters*, 26(2):277–281.

Ozaki, Yoshihiko, Yano, Masaki, and Onishi, Masaki. 2017. Effective hyperparameter optimization using Nelder-Mead method in deep learning. *IPSJ Transactions on Computer Vision and Applications*, 9(1):1–12.

Pantel, Patrick and Lin, Dekang. 2002. Discovering word senses from text. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, page 613–619, New York, NY, USA. Association for Computing Machinery.

Pappas, Thrasyvoulos N and Jayant, Nikil S. 1989. An adaptive clustering algorithm for image segmentation. In *International Conference on Acoustics, Speech, and Signal Processing,*, pages 1667–1670. IEEE.

Parikh, Ankur, Täckström, Oscar, Das, Dipanjan, and Uszkoreit, Jakob. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Park, Chanyoung, Kim, Donghyun, Han, Jiawei, and Yu, Hwanjo. 2020. Unsupervised attributed multiplex network embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5371–5378.

Pazzani, Michael J and Billsus, Daniel. 2007. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer.

Pearson, Karl. 1896. VII. mathematical contributions to the theory of evolution.–III. regression, heredity, and panmixia. *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, (187):253–318.

Peng, Shuai, Yuan, Ke, Gao, Liangcai, and Tang, Zhi. 2021. MathBERT: A pre-trained model for mathematical formula understanding. *CoRR*, abs/2105.00377.

Peng, Xi, Zhu, Hongyuan, Feng, Jiashi, Shen, Chunhua, Zhang, Haixian, and Zhou, Joey Tianyi. 2019. Deep clustering with sample-assignment invariance prior. *IEEE transactions on neural networks and learning systems*, 31(11):4857–4868.

Pennington, Jeffrey, Socher, Richard, and Manning, Christopher. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Peters, Matthew, Neumann, Mark, Zettlemoyer, Luke, and Yih, Wen-tau. 2018a. Dissecting Contextual Word Embeddings: Architecture and Representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.

Peters, Matthew E., Neumann, Mark, Iyyer, Mohit, Gardner, Matt, Clark, Christopher, Lee, Kenton, and Zettlemoyer, Luke. 2018b. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Pham, Thang, Bui, Trung, Mai, Long, and Nguyen, Anh. 2021. Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1145–1160, Online. Association for Computational Linguistics.

Pu, Guo, Wang, Lijuan, Shen, Jun, and Dong, Fang. 2021. A hybrid unsupervised clustering-based anomaly detection method. *Tsinghua Science and Technology*, 26(2):146–153.

Purohit, Harsh, Tanabe, Ryo, Endo, Takashi, Suefusa, Kaori, Nikaido, Yuki, and Kawaguchi, Yohei. 2020. Deep autoencoding GMM-based unsupervised anomaly detection in acoustic signals and its hyper-parameter optimization. *arXiv preprint arXiv:2009.12042*.

Qannari, El Mostafa, Wakeling, Ian, Courcoux, Philippe, and MacFie, Halliday JH. 2000. Defining the underlying sensory dimensions. *Food Quality and Preference*, 11(1-2):151–154.

Qiu, Chen, Pfrommer, Timo, Kloft, Marius, Mandt, Stephan, and Rudolph, Maja. 2021. Neural transformation learning for deep anomaly detection beyond images. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8703–8714. PMLR.

Quafafou, Mohamed and Boussouf, Moussa. 2000. Generalized rough sets based feature selection. *Intelligent Data Analysis*, 4(1):3–17.

Radford, Alec, Narasimhan, Karthik, Salimans, Tim, and Sutskever, Ilya. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Radford, Alec, Wu, Jeffrey, Child, Rewon, Luan, David, Amodei, Dario, Sutskever, Ilya, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Raffel, Colin, Shazeer, Noam, Roberts, Adam, Lee, Katherine, Narang, Sharan, Matena, Michael, Zhou, Yanqi, Li, Wei, and Liu, Peter J. 2020. Exploring the limits of transfer learning with a unified text-to-text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Raganato, Alessandro, Delli Bovi, Claudio, and Navigli, Roberto. 2017. Neural sequence learning models for word sense disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1156–1167, Copenhagen, Denmark. Association for Computational Linguistics.

Rajaee, Sara and Pilehvar, Mohammad Taher. 2021. How does fine-tuning affect the geometry of embedding space: A case study on isotropy. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3042–3049, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rajaee, Sara and Pilehvar, Mohammad Taher. 2022. An isotropy analysis in the multilingual BERT embedding space. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1309–1316, Dublin, Ireland. Association for Computational Linguistics.

Ramachandran, Prajit, Liu, Peter, and Le, Quoc. 2017. Unsupervised pretraining for sequence to sequence learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 383–391, Copenhagen, Denmark. Association for Computational Linguistics.

Ramaswamy, Sridhar, Rastogi, Rajeev, and Shim, Kyuseok. 2000. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, page 427–438, New York, NY, USA. Association for Computing Machinery.

Ranasinghe, Tharindu, Orasan, Constantin, and Mitkov, Ruslan. 2019. Enhancing unsupervised sentence similarity methods with deep contextualised word representations. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 994–1003, Varna, Bulgaria. INCOMA Ltd.

Raunak, Vikas, Gupta, Vivek, and Metze, Florian. 2019. Effective dimensionality reduction for word embeddings. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 235–243, Florence, Italy. Association for Computational Linguistics.

Rdusseeun, LKPJ and Kaufman, P. 1987. Clustering by means of medoids. In *Proceedings of the statistical data analysis based on the L1 norm conference, neuchatel, switzerland*, volume 31.

Rebbapragada, Umaa, Protopapas, Pavlos, Brodley, Carla E, and Alcock, Charles. 2009. Finding anomalous periodic time series. *Machine learning*, 74(3):281–313.

Reichardt, Jörg and Bornholdt, Stefan. 2006. Statistical mechanics of community detection. *Physical review E*, 74(1):016110.

Reimers, Nils and Gurevych, Iryna. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3980–3990, Hong Kong, China. Association for Computational Linguistics.

Richards, Joseph W, Freeman, Peter E, Lee, Ann B, and Schafer, Chad M. 2009. Exploiting low-dimensional structure in astronomical spectra. *The Astrophysical Journal*, 691(1):32.

Riverain, Paul, Fossier, Simon, and Nadif, Mohamed. 2022a. Poisson degree corrected dynamic stochastic block model. *Advances in Data Analysis and Classification*, pages 1–28.

Riverain, Paul, Fossier, Simon, and Nadif, Mohamed. 2022b. Semi-supervised latent block model with pairwise constraints. *Machine Learning*, 111(5):1739–1764.

Robert, Paul and Escoufier, Yves. 1976. A unifying tool for linear multivariate statistical methods: the RV-coefficient. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 25(3):257–265.

Rogers, Anna, Kovaleva, Olga, and Rumshisky, Anna. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Rohde, Douglas LT, Gonnerman, Laura M, and Plaut, David C. 2006. An improved model of semantic similarity based on lexical co-occurrence. *Communications of the ACM*, 8(627-633):116.

Rousseeuw, Peter J. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65.

Roweis, Sam T and Saul, Lawrence K. 2000. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326.

Rücklé, Andreas, Eger, Steffen, Peyrard, Maxime, and Gurevych, Iryna. 2018. Concatenated p-mean word embeddings as universal cross-lingual sentence representations. *CoRR*, abs/1803.01400.

Ruff, Lukas, Vandermeulen, Robert, Goernitz, Nico, Deecke, Lucas, Siddiqui, Shoaib Ahmed, Binder, Alexander, Müller, Emmanuel, and Kloft, Marius. 2018. Deep one-class classification. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4393–4402. PMLR.

Ruff, Lukas, Zemlyanskiy, Yury, Vandermeulen, Robert, Schnake, Thomas, and Kloft, Marius. 2019. Self-attentive, multi-context one-class classification for unsupervised anomaly detection on text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4061–4071, Florence, Italy. Association for Computational Linguistics.

Rumelhart, David E, Hinton, Geoffrey E, and Williams, Ronald J. 1985. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Sainburg, Tim, McInnes, Leland, and Gentner, Timothy Q. 2021. Parametric UMAP embeddings for representation and semisupervised learning. *Neural Computation*, 33(11):2881–2907.

Salah, Aghiles, Ailem, Melissa, and Nadif, Mohamed. 2018. Word co-occurrence regularized non-negative matrix tri-factorization for text data co-clustering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Salah, Aghiles and Nadif, Mohamed. 2017a. Model-based von Mises-Fisher co-clustering with a conscience. In *Proceedings of the 2017 SIAM International Conference on Data Mining (SDM)*, pages 246–254. SIAM.

Salah, Aghiles and Nadif, Mohamed. 2017b. Social regularized von Mises–Fisher mixture model for item recommendation. *Data Mining and Knowledge Discovery*, 31:1218–1241.

Salah, Aghiles and Nadif, Mohamed. 2019. Directional co-clustering. *Advances in Data Analysis and Classification*, 13:591–620.

Sanh, Victor, Debut, Lysandre, Chaumond, Julien, and Wolf, Thomas. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Sarwar, Badrul, Karypis, George, Konstan, Joseph, and Riedl, John. 2000. Application of dimensionality reduction in recommender system-a case study. Technical report, Minnesota Univ Minneapolis Dept of Computer Science.

Saxena, Amit, Prasad, Mukesh, Gupta, Akshansh, Bharill, Neha, Patel, Om Prakash, Tiwari, Aruna, Er, Meng Joo, Ding, Weiping, and Lin, Chin-Teng. 2017. A review of clustering techniques and developments. *Neurocomputing*, 267:664–681.

Schlegl, Thomas, Seeböck, Philipp, Waldstein, Sebastian M, Langs, Georg, and Schmidt-Erfurth, Ursula. 2019. f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks. *Medical image analysis*, 54:30–44.

Schmidhuber, Jürgen. 2015. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117.

Schölkopf, Bernhard, Mika, Sebastian, Smola, Alex, Rätsch, Gunnar, and Müller, Klaus-Robert. 1998a. Kernel PCA pattern reconstruction via approximate pre-images. In *International Conference on Artificial Neural Networks*, pages 147–152. Springer.

Schölkopf, Bernhard, Smola, Alexander, and Müller, Klaus-Robert. 1998b. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319.

Schölkopf, Bernhard, Williamson, Robert C, Smola, Alex, Shawe-Taylor, John, and Platt, John. 1999. Support vector method for novelty detection. *Advances in neural information processing systems*, 12.

Schreyer, Marco, Sattarov, Timur, Borth, Damian, Dengel, Andreas, and Reimer, Bernd. 2017. Detection of anomalies in large scale accounting data using deep autoencoder networks. *CoRR*, abs/1709.05254.

Schubert, Erich, Spitz, Andreas, Weiler, Michael, Geiß, Johanna, and Gertz, Michael. 2017. Semantic word clouds with background corpus normalization and t-distributed stochastic neighbor embedding.

Schölkopf, Bernhard, Platt, John C., Shawe-Taylor, John, Smola, Alex J., and Williamson, Robert C. 2001. Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 13(7):1443–1471.

Scialom, Thomas, Dray, Paul-Alexis, Lamprier, Sylvain, Piwowarski, Benjamin, and Staiano, Jacopo. 2020. MLSUM: The multilingual summarization corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8051–8067.

Sedding, Julian and Kazakov, Dimitar. 2004. Wordnet-based text document clustering. In *Proceedings of the 3rd workshop on RObust Methods in Analysis of Natural Language Data (ROMAND 2004)*, pages 104–113.

Shaoul, Cyrus and Westbury, Chris. 2006. Word frequency effects in high-dimensional co-occurrence models: A new approach. *Behavior Research Methods*, 38(2):190–195.

Sheikholeslami, Gholamhosein, Chatterjee, Surojit, and Zhang, Aidong. 1998. Wavecluster: A multi-resolution clustering approach for very large spatial databases. In *VLDB*, volume 98, pages 428–439.

Sheridan, Kevin, Puranik, Tejas G., Mangortey, Eugene, Pinon-Fischer, Olivia J., Kirby, Michelle, and Mavris, Dimitri N. An application of dbscan clustering for flight anomaly detection during the approach phase. In *AIAA Scitech 2020 Forum*.

Shi, Jianbo and Malik, Jitendra. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905.

Sibson, Robin. 1973. SLINK: an optimally efficient algorithm for the single-link cluster method. *The computer journal*, 16(1):30–34.

Siedlecki, W. and Sklansky, J. 1989. A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10(5):335–347.

Sinha, Koustuv, Parthasarathi, Prasanna, Pineau, Joelle, and Williams, Adina. 2021. UnNatural Language Inference. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7329–7346, Online. Association for Computational Linguistics.

Sokal, Robert R. 1958. A statistical method for evaluating systematic relationships. *Univ. Kansas, Sci. Bull.*, 38:1409–1438.

Song, Kaitao, Tan, Xu, Qin, Tao, Lu, Jianfeng, and Liu, Tie-Yan. 2019. MASS: Masked sequence to sequence pre-training for language generation. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5926–5936. PMLR.

Sorensen, Thorvald A. 1948. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on danish commons. *Biol. Skar.*, 5:1–34.

Spearman, Charles. 1987. The proof and measurement of association between two things. *The American journal of psychology*, 100(3/4):441–471.

Stears, Robin L, Martinsky, Todd, and Schena, Mark. 2003. Trends in microarray analysis. *Nature medicine*, 9(1):140–145.

Steinley, Douglas. 2004. Properties of the Hubert-Arable Adjusted Rand Index. *Psychological methods*, 9(3):386.

Strehl, Alexander and Ghosh, Joydeep. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617.

Su, Jianlin, Cao, Jiarun, Liu, Weijie, and Ou, Yangyiwen. 2021. Whitening sentence representations for better semantics and faster retrieval. *CoRR*, abs/2103.15316.

Subramanian, Sandeep, Trischler, Adam, Bengio, Yoshua, and Pal, Christopher J. 2018. Learning general purpose distributed sentence representations via large scale multi-task learning. In *International Conference on Learning Representations*.

Sun, Shiquan, Zhu, Jiaqiang, Ma, Ying, and Zhou, Xiang. 2019. Accuracy, robustness and scalability of dimensionality reduction methods for single-cell RNA-seq analysis. *Genome biology*, 20(1):1–21.

Takagishi, Mariko, van de Velden, Michel, and Yadohisa, Hiroshi. 2019. Clustering preference data in the presence of response-style bias. *British Journal of Mathematical and Statistical Psychology*, 72(3):401–425.

Tang, Yun and Rose, Richard. 2008. A study of using locality preserving projections for feature extraction in speech recognition. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1569–1572. IEEE.

Tao, Yaling, Takagi, Kentaro, and Nakata, Kouta. 2021. Clustering-friendly representation learning via instance discrimination and feature decorrelation. In *International Conference on Learning Representations*.

Tenenbaum, Joshua B, Silva, Vin de, and Langford, John C. 2000. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323.

Tenney, Ian, Das, Dipanjan, and Pavlick, Ellie. 2019a. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Tenney, Ian, Xia, Patrick, Chen, Berlin, Wang, Alex, Poliak, Adam, McCoy, R Thomas, Kim, Najoung, Durme, Benjamin Van, Bowman, Sam, Das, Dipanjan, and Pavlick, Ellie. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Tian, Fei, Gao, Bin, Cui, Qing, Chen, Enhong, and Liu, Tie-Yan. 2014. Learning deep representations for graph clustering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 28(1).

Tian, Kai, Zhou, Shuigeng, and Guan, Jihong. 2017. DeepCluster: A general clustering framework based on deep learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 809–825, Cham. Springer International Publishing.

Tian, Tian, Wan, Ji, Song, Qi, and Wei, Zhi. 2019. Clustering single-cell RNA-seq data with a model-based deep learning approach. *Nature Machine Intelligence*, 1(4):191–198.

Timmerman, Marieke E, Ceulemans, Eva, De Roover, Kim, and Van Leeuwen, Karla. 2013. Subspace k-means clustering. *Behavior research methods*, 45(4):1011–1023.

Tipping, Michael E and Bishop, Christopher M. 1999. Mixtures of probabilistic principal component analyzers. *Neural computation*, 11(2):443–482.

Torkkola, Kari. 2003. Feature extraction by non-parametric mutual information maximization. *Journal of machine learning research*, 3(Mar):1415–1438.

Tsai, Jeng-Tsung, Lin, Yen-Yu, and Liao, Hong-Yuan Mark. 2014. Per-cluster ensemble kernel learning for multi-modal image clustering with group-dependent feature selection. *IEEE Transactions on Multimedia*, 16(8):2229–2241.

Tucker, Ledyard R. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311.

Turney, Peter D and Pantel, Patrick. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188.

Uria, Benigno, Côté, Marc-Alexandre, Gregor, Karol, Murray, Iain, and Larochelle, Hugo. 2016. Neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 17(205):1–37.

Uysal, Alper Kursat. 2018. On two-stage feature selection methods for text classification. *IEEE Access*, 6:43233–43251.

van Aken, Betty, Winter, Benjamin, Löser, Alexander, and Gers, Felix A. 2019. How Does BERT Answer Questions?: A Layer-Wise Analysis of Transformer Representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1823–1832, Beijing China. ACM.

van Buuren, Stef and Heiser, Willem J. 1989. Clustering n objects into k groups under optimal scaling of variables. *Psychometrika*, 54(4):699–706.

van Dam, Jan-Willem and van de Velden, Michel. 2015. Online profiling and clustering of Facebook users. *Decision Support Systems*, 70:60–72.

van de Velden, Michel, Iodice D'Enza, Alfonso, and Markos, Angelos. 2019. Distance-based clustering of mixed data. *Wiley Interdisciplinary Reviews: Computational Statistics*, 11(3):e1456.

van der Maaten, Laurens. 2013. Barnes-Hut-SNE. *arXiv preprint arXiv:1301.3342*.

van der Maaten, Laurens and Hinton, Geoffrey. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605.

van der Maaten, Laurens, Postma, Eric, and van den Herik, Jaap. 2009. Dimensionality reduction: A comparative review. Technical report, TiCC-TR 2009-005, Tilburg University.

Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Łukasz, and Polosukhin, Illia. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Vega-Pons, Sandro and Ruiz-Shulcloper, José. 2011. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(03):337–372.

Vial, Loïc, Lecouteux, Benjamin, and Schwab, Didier. 2018. UFSAC: Unification of sense annotated corpora and tools. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Vichi, Maurizio and Kiers, Henk AL. 2001. Factorial k-means analysis for two-way data. *Computational Statistics & Data Analysis*, 37(1):49–64.

Vincent, Pascal, Larochelle, Hugo, Lajoie, Isabelle, Bengio, Yoshua, and Manzagol, Pierre-Antoine. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(110):3371–3408.

Von Luxburg, Ulrike. 2007. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416.

Vulić, Ivan, Ponti, Edoardo Maria, Litschko, Robert, Glavaš, Goran, and Korhonen, Anna. 2020. Probing Pretrained Language Models for Lexical Semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Wang, Guan, Xie, Sihong, Liu, Bing, and Philip, S Yu. 2011. Review graph based online store review spammer detection. In *2011 IEEE 11th international conference on data mining*, pages 1242–1247. IEEE.

Wang, Hongzhi, Bah, Mohamed Jaward, and Hammad, Mohamed. 2019. Progress in outlier detection techniques: A survey. *IEEE Access*, 7:107964–108000.

Wang, Lingxiao, Huang, Jing, Huang, Kevin, Hu, Ziniu, Wang, Guangtao, and Gu, Quanquan. 2020. Improving neural language generation with spectrum control. In *International Conference on Learning Representations*.

Ward Jr, Joe H. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

Wen, Yeming, Tran, Dustin, and Ba, Jimmy. 2020. BatchEnsemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*.

White, Scott and Smyth, Padhraic. A spectral clustering approach to finding communities in graphs. In *Proceedings of the 2005 SIAM International Conference on Data Mining (SDM)*, pages 274–285.

Wilderjans, Tom F. and Cariou, Véronique. 2016. CLV3W: A clustering around latent variables approach to detect panel disagreement in three-way conventional sensory profiling data. *Food Quality and Preference*, 47:45–53. Sensometric 2014: Data That Works In The City That Works.

Williams, Christopher. 2000. On a connection between kernel PCA and metric multidimensional scaling. *Advances in neural information processing systems*, 13.

Williams, Trefor P and Gong, Jie. 2014. Predicting construction cost overruns using text mining, numerical data and ensemble classifiers. *Automation in Construction*, 43:23–29.

Wold, S, Hellberg, S, Lundstedt, T, Sjöström, M, and Wold, H. 1987. PLS model building: Theory and application. PLS modeling with latent variables in two or more dimensions. In *PLS Symposium. Frankfurt*.

Wu, Zhenyu and Leahy, Richard. 1993. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 15(11):1101–1113.

Xiang, Tao and Gong, Shaogang. 2005. Video behaviour profiling and abnormality detection without manual labelling. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1238–1245. IEEE.

Xiao, Han. 2018. bert-as-service. https://github.com/hanxiao/bert-as-service.

Xie, Junyuan, Girshick, Ross, and Farhadi, Ali. 2016. Unsupervised deep embedding for clustering analysis. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 478–487, New York, New York, USA. PMLR.

Xu, Huan, Caramanis, Constantine, and Sanghavi, Sujay. 2010. Robust PCA via outlier pursuit. *Advances in neural information processing systems*, 23.

Xu, Jiaming, Wang, Peng, Tian, Guanhua, Xu, Bo, Zhao, Jun, Wang, Fangyuan, and Hao, Hongwei. 2015. Short text clustering via convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 62–69, Denver, Colorado. Association for Computational Linguistics.

Xu, Jiaming, Xu, Bo, Wang, Peng, Zheng, Suncong, Tian, Guanhua, and Zhao, Jun. 2017. Self-taught convolutional neural networks for short text clustering. *Neural Networks*, 88:22–31.

Xu, Rongbin, Che, Yan, Wang, Xinmei, Hu, Jianxiong, and Xie, Ying. 2020. Stacked autoencoder-based community detection method via an ensemble clustering framework. *Information Sciences*, 526:151–165.

Xu, Rui and Wunsch, Donald. 2005. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678.

Xu, Wei and Gong, Yihong. 2004. Document clustering by concept factorization. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, page 202–209, New York, NY, USA. Association for Computing Machinery.

Xu, Wei, Liu, Xin, and Gong, Yihong. 2003. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, page 267–273, New York, NY, USA. Association for Computing Machinery.

Xue, Linting, Constant, Noah, Roberts, Adam, Kale, Mihir, Al-Rfou, Rami, Siddhant, Aditya, Barua, Aditya, and Raffel, Colin. 2021. mT5: A massively multilingual pre-trained text-to-text Transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Yan, Yuanmeng, Li, Rumei, Wang, Sirui, Zhang, Fuzheng, Wu, Wei, and Xu, Weiran. 2021. ConSERT: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5065–5075, Online. Association for Computational Linguistics.

Yang, Bo, Fu, Xiao, Sidiropoulos, Nicholas D., and Hong, Mingyi. 2017. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3861–3870. PMLR.

Yang, Li and Shami, Abdallah. 2020. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316.

Yang, Liang, Cao, Xiaochun, He, Dongxiao, Wang, Chuan, Wang, Xiao, and Zhang, Weixiong. 2016. Modularity based community detection with deep learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 2252–2258. AAAI Press.

Yang, Linxiao, Cheung, Ngai-Man, Li, Jiaying, and Fang, Jun. 2019a. Deep clustering by Gaussian mixture variational autoencoders with graph embedding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6440–6449.

Yang, Yi, Uy, Mark Christopher Siy, and Huang, Allen. 2020. FinBERT: A pretrained language model for financial communications. *CoRR*, abs/2006.08097.

Yang, Zhilin, Dai, Zihang, Yang, Yiming, Carbonell, Jaime, Salakhutdinov, Russ R, and Le, Quoc V. 2019b. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Yau, Christopher et al. 2016. pcaReduce: hierarchical clustering of single cell transcriptional profiles. *BMC bioinformatics*, 17(1):1–11.

Yoo, Illhoi, Alafaireet, Patricia, Marinov, Miroslav, Pena-Hernandez, Keila, Gopidi, Rajitha, Chang, Jia-Fu, and Hua, Lei. 2012. Data mining in healthcare and biomedicine: a survey of the literature. *Journal of medical systems*, 36(4):2431–2448.

Yoo, Illhoi and Hu, Xiaohua. 2006. A comprehensive comparison study of document clustering for a biomedical digital library MEDLINE. In *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 220–229.

Yu, Chun, Chen, Kun, and Yao, Weixin. 2015a. Outlier detection and robust mixture modeling using nonconvex penalized likelihood. *Journal of Statistical Planning and Inference*, 164:27–38.

Yu, Rose, He, Xinran, and Liu, Yan. 2015b. GLAD: Group anomaly detection in social media analysis. *ACM Transactions on Knowledge Discovery from Data*, 10(2):1–22.

Yuan, Ming and Huang, Jianhua Z. 2009. Regularized parameter estimation of high dimensional t distribution. *Journal of Statistical Planning and Inference*, 139(7):2284–2292.

Yun, Zeyu, Chen, Yubei, Olshausen, Bruno, and LeCun, Yann. 2021. Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of Transformer factors. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 1–10, Online. Association for Computational Linguistics.

Zhang, Tian, Ramakrishnan, Raghu, and Livny, Miron. 1996. BIRCH: An efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, SIGMOD '96, page 103–114, New York, NY, USA. Association for Computing Machinery.

Zhang, Tianyi, Kishore, Varsha, Wu, Felix, Weinberger, Kilian Q., and Artzi, Yoav. 2020a. BERTscore: Evaluating text generation with BERT. In *International Conference on Learning Representations*.

Zhang, Xiang, Zhao, Junbo, and LeCun, Yann. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Zhang, Yan, He, Ruidan, Liu, Zuozhu, Lim, Kwan Hui, and Bing, Lidong. 2020b. An unsupervised sentence embedding method by mutual information maximization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1601–1610, Online. Association for Computational Linguistics.

Zhao, Yue and Akoglu, Leman. 2022. Towards unsupervised HPO for outlier detection. *arXiv preprint arXiv:2208.11727*.

Zhao, Yue, Rossi, Ryan A., and Akoglu, Leman. 2020. Automating outlier detection via meta-learning. *CoRR*, abs/2009.10606.

Zhou, Sheng, Xu, Hongjia, Zheng, Zhuonan, Chen, Jiawei, Bu, Jiajun, Wu, Jia, Wang, Xin, Zhu, Wenwu, Ester, Martin, et al. 2022. A comprehensive survey on deep clustering: Taxonomy, challenges, and future directions. *arXiv preprint arXiv:2206.07579*.

Zhu, Xiaofeng, Suk, Heung-Il, Lee, Seong-Whan, and Shen, Dinggang. 2016. Subspace regularized sparse multitask learning for multiclass neurodegenerative disease identification. *IEEE Transactions on Biomedical Engineering*, 63(3):607–618.

Zhuang, Honglei, Wang, Chi, Tao, Fangbo, Kaplan, Lance, and Han, Jiawei. 2017. Identifying semantically deviating outlier documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2748–2757, Copenhagen, Denmark. Association for Computational Linguistics.

Zong, Bo, Song, Qi, Min, Martin Renqiang, Cheng, Wei, Lumezanu, Cristian, Cho, Daeki, and Chen, Haifeng. 2018. Deep autoencoding Gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*.

Zou, Hui and Xue, Lingzhou. 2018. A selective overview of sparse principal component analysis. *Proceedings of the IEEE*, 106(8):1311–1320.

Appendix

Appendix A. Additional Word-level Experiments

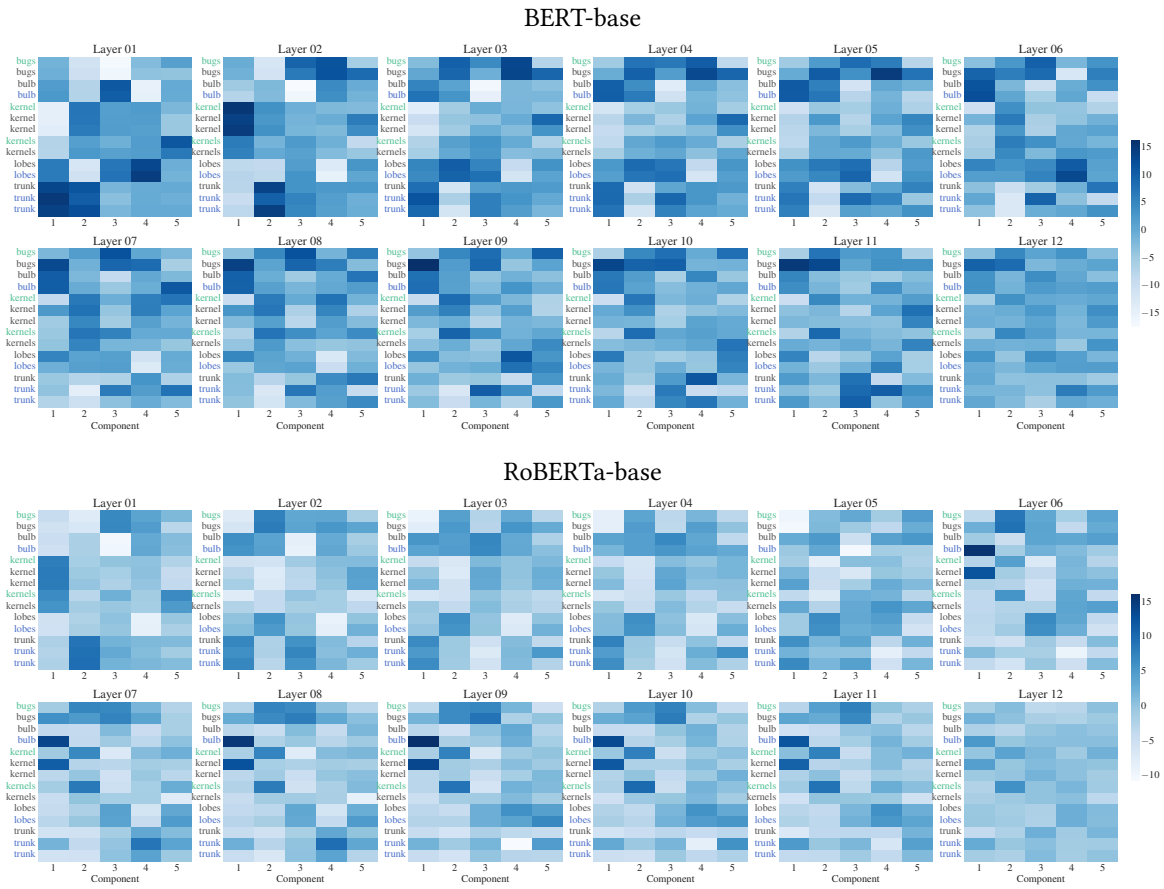

Figure A.1: Sensitivity to context: these graphics depict the 5 first principal components of 5 words appearing in different contexts. The color of the word determines the topic of the sentence it appears in.

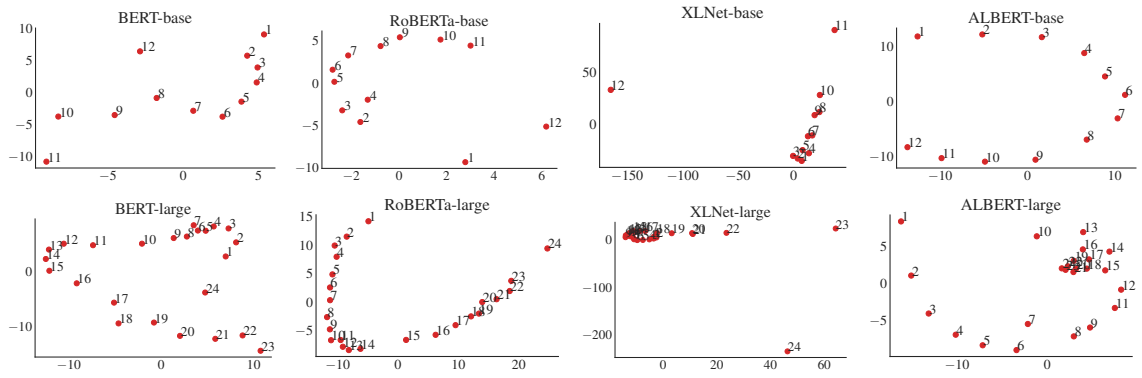

Figure A.2: Multidimensional scaling visualizations of the averaged word vectors of UFSAC4.

Appendix B. Additional Results for Chapter 4

Before proceeding to the partitioning of the representations from the Transformers, we analyze the two-dimensional compressions of the embeddings provided by the different layers of four models. For this, we use t-SNE (van der Maaten and Hinton, 2008) with the same parameters (perplexity at 15 and learning rate at 200) at each execution. We report the evolution of the layers in the figure B.1.

We can observe a variability of the structure in classes according to the layers. For example, for BERT-base, the separability of the four classes of classic4, where the classes are not very well separated, deteriorates when approaching the end of the network, with in particular the appearance of an additional class. However, the representations provided by RoBERTa seem to allow a better separability of classes in general compared to BERT, with better results, in particular, on the last layers. This suggests that, like PCA_w, t-SNE allows to alleviate the difficulty regarding the poor quality representations of RoBERTa-base.

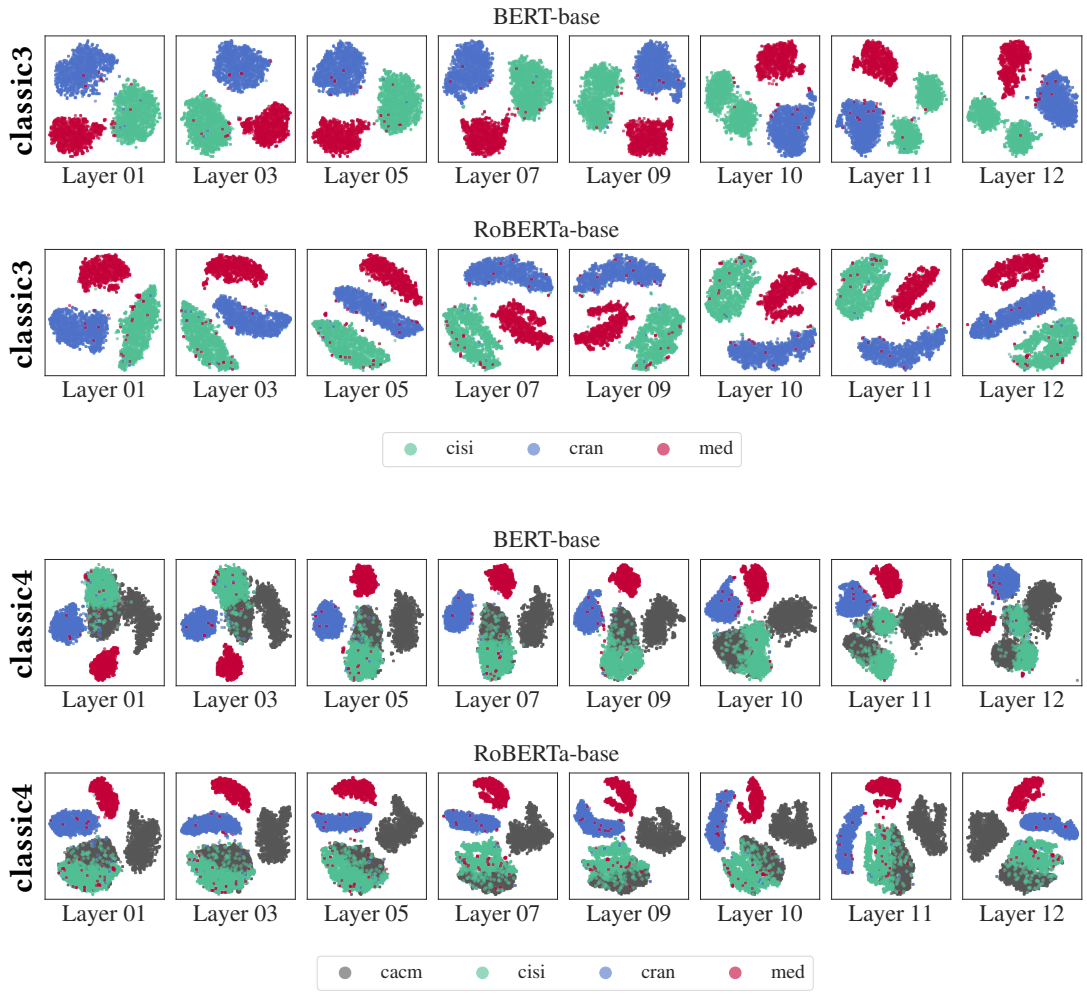

Figure B.1: Two-dimensional projections obtained with t-SNE. The colors correspond to the real class labels.

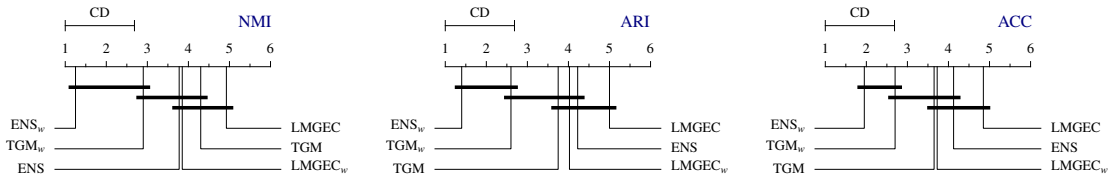

Figure B.2: Comparison between whitened and non-whitened versions of step-wise and end-to-end consensus approaches.

Appendix C. Additional Results for Chapter 5

Figure C.1 show the NMI score obtained with two post-processing strategies using different values of d'. It shows the power of post-processing on different representations when using

less than 10% of the dimensions (for BERT) and 30% for static representations. This graphic also shows the relative instability of PCA_w despite its effectiveness, especially on classic4 and BBC. One way of overcoming this instability is to rely on the ensemble approach (as seen in Chapter 4) that allows for reducing the impact of low-performance representations.

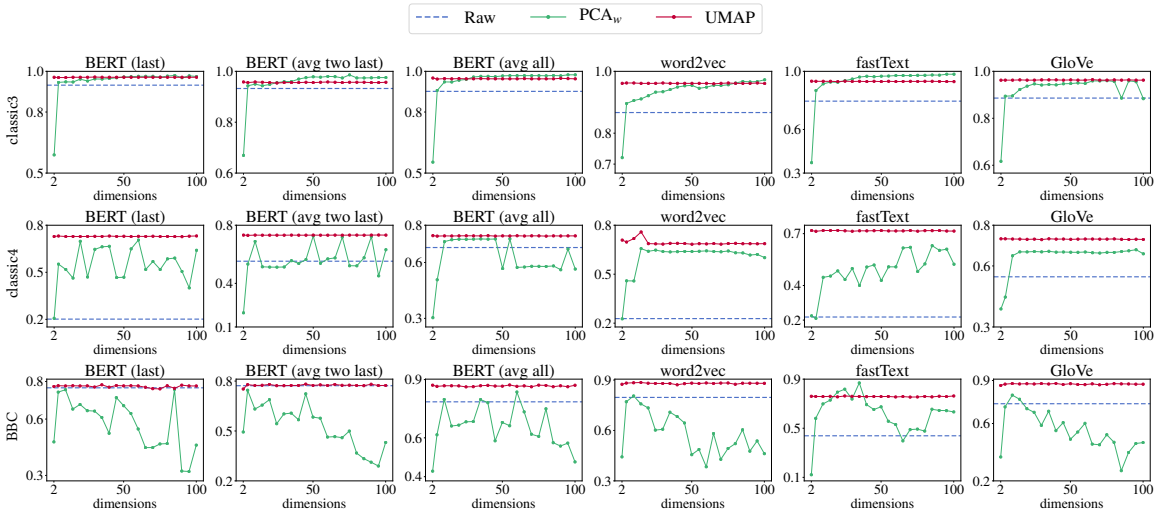

Figure C.1: Clustering performance with two post-processing strategies using different values of d'.

Table C.1: p-values of the pearson test of correlation between several isotropy and performance measures. The corresponding correlation coefficients are given in Table 5.4. Values under 10^{-3} are considered near-zero.

		NMI		ARI		Dataset			External (word)			External (sentence)		
		km	*skm*	*km*	*skm*	*cos*	*pf*	pf_2	*cos*	*pf*	pf_2	*cos*	*pf*	pf_2
NMI	*km*	0.0	≈ 0.0	≈ 0.0	≈ 0.0	0.647	0.858	0.644	0.225	0.88	0.86	0.54	0.964	0.896
	skm	≈ 0.0	0.0	≈ 0.0	≈ 0.0	0.453	0.662	0.195	0.642	0.64	0.658	0.476	0.616	0.635
ARI	*km*	≈ 0.0	≈ 0.0	0.0	≈ 0.0	0.562	0.921	0.542	0.297	0.946	0.925	0.542	0.98	0.96
	skm	≈ 0.0	≈ 0.0	≈ 0.0	0.0	0.934	0.71	0.517	0.56	0.742	0.722	0.877	0.755	0.741
Dataset	*cos*	0.647	0.453	0.562	0.934	0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0
	pf	0.858	0.662	0.921	0.71	≈ 0.0	0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0
	pf_2	0.644	0.195	0.542	0.517	≈ 0.0	≈ 0.0	0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0
Ext-w	*cos*	0.225	0.642	0.297	0.56	≈ 0.0	≈ 0.0	≈ 0.0	0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0
	pf	0.88	0.64	0.946	0.742	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0
	pf_2	0.86	0.658	0.925	0.722	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	0.0	≈ 0.0	≈ 0.0	≈ 0.0
Ext-s	*cos*	0.54	0.476	0.542	0.877	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	0.0	≈ 0.0	≈ 0.0
	pf	0.964	0.616	0.98	0.755	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	0.0	≈ 0.0
	pf_2	0.896	0.635	0.96	0.741	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	≈ 0.0	0.0