Extraction d'événements à partir de peu d'exemples par méta-apprentissage

Aboubacar Tuo

université PARIS-SACLAY

Extraction d'événements à partir de peu d'exemples par méta-apprentissage

Meta-Learning for Few-Shot Event Extraction

Thèse de doctorat de l'Université Paris-Saclay

École doctorale n°580, Sciences et Technologies de l'Information et de la Communication (STIC)
Spécialité de doctorat : Informatique
Graduate School : Informatique et Sciences du Numérique
Référent : Faculté des Sciences d'Orsay

Thèse préparée dans l'**Institut LIST (Université Paris-Saclay, CEA**), sous la direction de **Olivier FERRET**, Directeur de recherche, le co-encadrement de **Romaric BESANÇON** et de **Julien TOURILLE**, Ingénieurs chercheurs.

Thèse soutenue à Paris-Saclay, le 20 décembre 2023, par

Aboubacar TUO

Composition du jury

Membres du jury avec voix délibérative

Claire NÉDELLEC Directrice de recherche, Université Paris-Saclay	Présidente
Vincent CLAVEAU Chargé de recherche, HDR, CNRS / IRISA Rennes	Rapporteur & Examinateur
Christophe GRAVIER Professeur, Université Jean Monnet	Rapporteur & Examinateur
Matthieu LABEAU Maître de conférences, Institut Polytechnique de Paris	Examinateur

THESE DE DOCTORAT
NNT : 2023UPASG098

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Extraction d'événements à partir de peu d'exemples par méta-apprentissage
Mots clés : Extraction d'information, Extraction d'événements, Apprentissage à partir de peu d'exemples, Méta-apprentissage, Apprentissage de représentations, Injection de connaissances

Résumé : L'extraction d'information est un champ de recherche dont l'objectif est d'identifier et extraire automatiquement des informations structurées, dans un domaine donné, à partir de données textuelles pas ou peu structurées. La mise en œuvre de telles extractions demande souvent des moyens humains importants pour l'élaboration de règles d'extraction ou encore pour la constitution de données annotées pour les systèmes utilisant de l'apprentissage automatique. Un des défis actuels dans le domaine de l'extraction d'information est donc de développer des méthodes permettant de réduire, dans la mesure du possible, les coûts et le temps de développement de ces systèmes. Ce travail de thèse se concentre sur l'exploration de l'extraction d'événements à travers l'utilisation du méta-apprentissage, une approche adaptée à l'apprentissage à partir de peu de données. Nous avons redéfini la tâche d'extraction d'événements dans cette perspective, cherchant à développer des systèmes capables de s'adapter rapidement à de nouveaux contextes d'extraction avec un faible volume de données d'entraînement. Dans un premier temps, nous avons proposé des méthodes visant à améliorer la détection des déclencheurs événementiels en développant des représentations plus robustes pour cette tâche. Ensuite, nous avons abordé le défi spécifique posé par la classe « *NULLE* » (absence d'événement) dans ce cadre. Enfin, nous avons évalué l'effectivité de nos propositions dans le contexte global de l'extraction d'événements en les étendant à l'extraction des arguments des événements.

Title : Meta-Learning for Few-Shot Event Extraction
Keywords : Information Extraction, Event Extraction, Few-Shot Learning, Meta Learning, Representation Learning, Knowledge Injection.

Abstract : Information Extraction (IE) is a research field with the objective of automatically identifying and extracting structured information within a given domain from unstructured or minimally structured text data. The implementation of such extractions often requires significant human efforts, either in the form of rule development or the creation of annotated data for systems based on machine learning. One of the current challenges in information extraction is to develop methods that minimize the costs and development time of these systems whenever possible. This thesis focuses on few-shot event extraction through a meta-learning approach that aims to train IE models from only few data. We have redefined the task of event extraction from this perspective, aiming to develop systems capable of quickly adapting to new contexts with a small volume of training data. First, we propose methods to enhance event trigger detection by developing more robust representations for this task. Then, we tackle the specific challenge raised by the "NULL" class (absence of events) within this framework. Finally, we evaluate the effectiveness of our proposals within the broader context of event extraction by extending their application to the extraction of event arguments.

Remerciements

Avant tout propos, je souhaite adresser mes remerciements et toute ma gratitude aux personnes qui ont contribué et aidé à la réalisation de ce travail.

Je tiens tout d'abord à adresser mes sincères remerciements à mon jury de thèse : Madame Claire NÉDELLEC et Messieurs Vincent CLAVEAU, Christophe GRAVIER et Matthieu LABEAU, pour leur précieuse évaluation de mon travail et leurs commentaires constructifs qui ont contribué à son amélioration. Double merci à M. LABEAU qui a également évalué mon travail à mi-parcours au côté de Monsieur Pierre ZWEIGENBAUM que je remercie également.

Je souhaite ensuite exprimer ma profonde gratitude envers mes encadrants : Monsieur Olivier FERRET - le directeur de thèse, et Messieurs Romaric BESANÇON et Julein TOURILLE - les co-encadrants, pour leur confiance et leur soutien constant. Votre expertise et vos conseils avisés m'ont guidé tout au long de cette thèse. Trois personnalités différentes, trois expertises différentes et trois expériences différentes, je ne pouvais pas espérer mieux !
Je suis également reconnaissant envers Madame Bianca VIERU, cheffe du laboratoire LASTI, pour sa confiance et son soutien continu dans mes démarches administratives.

Un grand merci à mes collègues, en particulier Monsieur Hervé LE BORGNE, pour son assistance technique à mon arrivée et son expertise scientifique quand j'en avais besoin ; et Madame Tiphaine LE CLERCQ DE LANNOY pour nos nombreuses discussions autour de nos problèmes scientifiques communs, ainsi que pour nos moments de rire partagés. Je tiens à exprimer ma reconnaissance envers les doctorants du Laboratoire LASTI qui ont grandement enrichi mon parcours de thèse, en particulier Evan DUFRAISSE et Michael SOUMM qui sont devenus de véritables amis aujourd'hui.

Mes remerciements vont également à ma famille et à mes amis pour leur soutien moral et leurs encouragements dans les moments de doute.
Enfin, je tiens à adresser un immense merci à Romane, l'amour de ma vie, ainsi qu'à sa famille, pour leur bienveillance et leur soutien indéfectible tout au long de cette aventure.

Votre présence a été essentielle dans la réalisation de cette thèse, et je vous en suis, tous et toutes, profondément reconnaissant.

Table des matières

Liste des tableaux

Table des figures

Chapitre 1

Introduction

Sommaire

L'explosion de données textuelles disponibles sur le Web a engendré un besoin croissant de méthodes d'analyse automatique de ces données. Dans ce contexte, l'Extraction d'information à partir de textes est devenue un domaine de recherche très actif visant à transformer ces données textuelles en données structurées et exploitables (Grishman, 2019). Elle comprend plusieurs sous-tâches de difficulté variable, telles que la Reconnaissance d'Entités Nommées (NER), la Résolution de Coréférence [1], l'Extraction de Relations (RE) et l'Extraction d'Événements (EE). La figure 1.1 donne une vue d'ensemble de ces différentes sous-tâches.

La **Reconnaissance d'Entités Nommées** est une tâche consistant à identifier et classer les mentions d'entités spécifiques dans un texte. Les entités nommées font référence à un certain nombre d'unités lexicales particulières, qui peuvent être les noms de personnes, d'organisation et de lieux, ensemble auquel sont souvent ajoutés d'autres syntagmes comme les dates, les unités monétaires, les pourcentages, etc. L'objectif de la tâche de NER est d'extraire ces entités et de les catégoriser dans des classes prédéfinies, permettant ainsi d'identifier les briques de base caractérisant l'information contenue dans le texte pour ensuite la structurer.

1. La résolution de coréférences, bien qu'elle ne soit pas spécifique à l'extraction d'information, est souvent associée à cette dernière, car elle permet d'identifier les relations entre différentes mentions d'un même élément, favorisant ainsi une meilleure compréhension des textes.

La **Résolution de Coréférence** vise à identifier et à regrouper les différentes mentions d'une même entité dans un texte. Lorsqu'un texte fait référence à une entité spécifique (par exemple, une personne, un objet ou un concept), il peut utiliser différents types d'expressions pour la désigner, tels que des pronoms ou des descriptions définies (par exemple, « le Président de la République »). La résolution de coréférence consiste à déterminer quelles expressions dans le texte se réfèrent à la même entité, permettant ainsi de comprendre et de relier ces informations de manière cohérente. Elle joue ainsi un rôle crucial dans la compréhension des textes et dans la construction de représentations structurées de l'information contenue dans ces textes.

L'**Extraction de Relations** vise à identifier et à extraire les relations entre différentes entités. Ces relations peuvent représenter des liens de divers types, tels que des relations de parenté, de localisation, d'appartenance, de causalité, etc. L'objectif de la tâche d'extraction de relations est de détecter et de classifier ces relations afin de mieux comprendre les interactions qui peuvent exister entre les entités nommées.

Enfin, l'**Extraction d'Événements** consiste à identifier, extraire et structurer les informations relatives aux événements dans du texte. Un événement peut être défini comme une action, une activité ou un processus qui se produit à un moment donné. L'objectif de l'extraction d'événements est de repérer les expressions linguistiques qui décrivent ces événements, d'identifier les participants ou les acteurs impliqués, les relations entre les événements et d'extraire les attributs associés tels que la date, le lieu, le temps. Les autres tâches d'extraction d'information constituent une étape préliminaire essentielle pour l'extraction d'événements, car elles permettent d'identifier les éléments pertinents (acteurs, lieux, dates, etc.) qui décrivent les événements.

Dans cette thèse, nous nous concentrons principalement sur la tâche d'extraction d'événements, à partir de données textuelles pas ou peu structurées, dans un contexte où on ne dispose que de peu de données annotées pour la définition de nouveaux événements à extraire.

1.1 . L'Extraction d'Événements

L'Extraction d'Événements est la forme la plus complexe des processus d'extraction d'informations, qui recouvre à la fois l'extraction des entités nommées et des relations qui les lient entre elles. Étant donné que la notion d'événement est spécifique à chaque domaine, il n'existe actuellement aucune définition stricte et précise d'un événement. Toutefois, le guide d'annotation de la campagne d'évaluation ACE-2005 (Doddington et al., 2004) (*Automatic Content Extraction, 2005*) en donne cette définition : « *An Event is a specific occurrence involving participants. An Event is something that happens. An Event can frequently be described as a change of state.* ». En d'autres termes, un événement peut être défini comme une occurrence spécifique à un moment et un lieu précis, impliquant

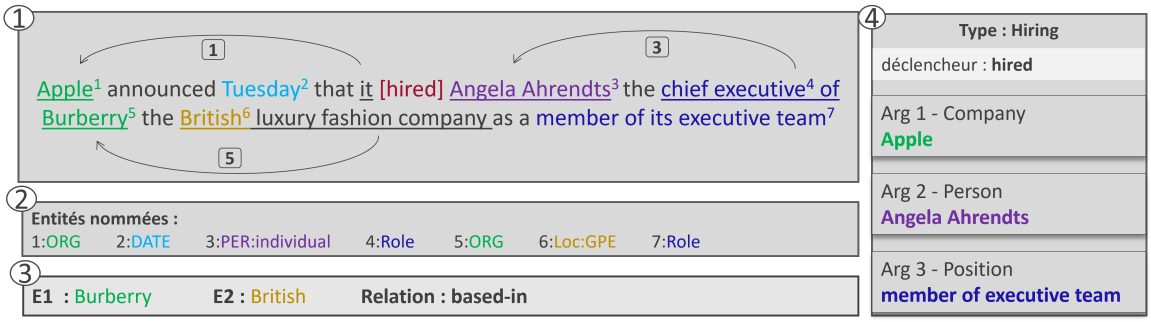


Figure 1.1 – Exemple de tâches d'extraction d'information : les mentions soulignées dans le cadre 1 représentent les entités nommées de la phrase dont le type est précisé dans le cadre 2, auxquelles s'ajoutent les mentions obtenues via les liens de coréférence (cf. flèches). Le cadre 3 concerne l'extraction de relations où l'on peut voir une relation d'origine géographique entre les entités Burberry et British. Le cadre 4 présente l'extraction d'événements en identifiant le déclencheur événementiel auquel on associe un type, puis les arguments sélectionnés parmi les entités identifiées.

un ou plusieurs participants et conduisant ou non à un changement d'état. L'extraction d'événements peut également être assimilée au remplissage, par des entités du texte, d'un formulaire dans lequel les champs correspondent aux arguments de l'événement (cf. exemple de la figure 1.1), comme considéré dans les campagnes d'évaluation MUC (*Message Understanding Conference* [2]).

Ces campagnes d'évaluation classent les événements en différents types, dont le nombre et le niveau de granularité peuvent varier en fonction des campagnes et des guides d'annotation. Chaque type d'événement est décrit de manière précise par des entités du texte appelées *Arguments* de l'événement, correspondant, aux champs du formulaire associé au type d'événement en question. Bien que l'objectif principal de l'extraction d'événements soit d'identifier les arguments d'un événement, les jeux de données depuis la campagne ACE-2005 ont introduit le concept de déclencheur événementiel (*Event Trigger*), désignant le groupe de mots indiquant le plus explicitement possible la présence d'un événement dans une phrase, le rôle de ce déclencheur étant de créer un ancrage lexical afin d'aider à la recherche des arguments par la suite. L'extraction d'événements peut dès lors être envisagée en deux étapes : d'abord, l'identification des déclencheurs et leur classification en fonction d'un ensemble de types d'événements prédéfinis, tâche appelée Détection d'Événements (*Event Detection, ED ou Trigger Detection*) ; puis l'identification des arguments de l'événement détecté et la classification de ces arguments en fonction de leur rôle vis-à-vis de l'événement, appelée Extraction des Arguments (*Event Argument Extraction, EAE*). L'extraction d'événements est dès lors divisée en deux sous-tâches principales, qui peuvent être réalisées de manière indépendante ou conjointe :

2. https ://www-nlpir.nist.gov/related_projects/muc/index.html

- **La détection d'événements :** qui consiste à identifier le déclencheur d'événement et lui associer un type. Cette tâche peut elle-même être décomposée en deux sous-tâches :
  - **Identification du déclencheur :** permettant d'identifier et localiser le déclencheur événementiel au sein d'un passage ;
  - **Classification d'événement :** consistant à assigner un type à une mention d'événement étant donné son déclencheur ;
- **L'extraction des arguments d'événements :** qui consiste à identifier les arguments et leur associer un rôle dans l'événement. De même, cette sous-tâche peut être décomposée en deux sous-tâches :
  - **L'identification des arguments :** consistant à identifier les arguments impliqués dans l'événement ; et
  - **La classification des arguments :** permettant d'assigner des rôles aux arguments identifiés.

Les premiers systèmes d'extraction d'événements reposaient sur des règles manuellement rédigées par des experts (Appelt and Onyshkevych, 1998; Cunningham et al., 2002; Levy and Andrew, 2006; Chiticariu et al., 2013; Valenzuela-Escárcega et al., 2016). Ces approches à base de règles sont généralement très précises et flexibles, mais elles nécessitent une expertise approfondie dans le domaine d'étude ainsi qu'une expertise technique et linguistique avancée pour leur formulation. Ainsi, des approches par apprentissage statistique ont progressivement pris le relais de ces méthodes à base de règles. Ces approches permettent soit d'apprendre automatiquement les règles d'extraction, soit d'apprendre directement la tâche à partir de caractéristiques (*features*) extraites des données textuelles.

Plus récemment, l'avènement de l'apprentissage profond (*Deep Learning*, DL) a entraîné un déclin progressif des méthodes par apprentissage statistique au profit des réseaux de neurones. Cette transition s'explique par le fait que l'apprentissage profond permet de s'affranchir à la fois de la nécessité d'écrire manuellement des règles et de l'identification manuelle des traits à prendre en compte.

Cependant, pour former des modèles neuronaux performants, il est généralement nécessaire d'avoir à la fois des données annotées de bonne qualité et en quantité suffisante. L'obtention de ces annotations est souvent coûteuse, en particulier dans des domaines spécialisés, où des experts du domaine sont requis pour annoter les ensembles de données. Par conséquent, quelle que soit l'approche envisagée (règles, apprentissage statistique ou apprentissage profond), la mise en place des systèmes d'extraction d'événements reste coûteuse en termes de ressources humaines. Les recherches récentes s'efforcent donc de réduire autant que possible ces coûts, afin de pouvoir généraliser rapidement ces modèles à de nouveaux types d'événements et s'adapter à différents

contextes applicatifs. Ces différences de contexte pouvant toucher le domaine (médecine, domaine légal, la finance, industrie...), le type des documents et des sources d'information considérés (articles scientifiques, articles de presse, réseaux sociaux), ou encore la langue de ces documents.

Le sujet de notre thèse s'inscrit directement dans cet objectif général en offrant la possibilité de définir des modèles d'extraction d'événements, pour un nouveau contexte applicatif, à partir de peu d'exemples.

1.2 . Problématiques

Dans cette thèse, nous nous intéressons à la tâche d'extraction d'événements, utilisant une approche par apprentissage profond, dans un contexte où les données d'entraînement sont limitées. Ce contexte est couramment appelé l'apprentissage à partir de peu d'exemples. Plus spécifiquement, nous nous concentrons sur l'utilisation du méta-apprentissage (*Meta-learning*) (Finn et al., 2017a; Ravi and Larochelle, 2016), qui a démontré son efficacité pour l'apprentissage à partir de peu d'exemples dans des tâches de vision par ordinateur telles que la classification d'images (Snell et al., 2017; Vinyals et al., 2016; Sung et al., 2018) ou la segmentation d'images (Wang et al., 2019a), ainsi que dans des tâches de Traitement Automatique des Langues (TAL), comme la classification de textes (Zhang et al., 2022a), la reconnaissance d'entités nommées (Fritzler et al., 2018) ou l'extraction de relations (Han et al., 2018). Le méta-apprentissage consiste à apprendre à partir de multiples tâches d'apprentissage afin de généraliser et adapter rapidement le modèle appris à de nouvelles tâches similaires. Une tâche au sens du méta-apprentissage est un épisode d'apprentissage qui simule un scénario d'apprentissage spécifique avec peu d'exemples. Ce paradigme d'apprentissage est parfois défini comme le fait « d'apprendre à apprendre ».

Nous formulons ici la tâche d'extraction d'événements comme un problème de classification multi-classes, où chaque classe représente un type d'événement. Cette approche nous permet de tirer parti des techniques de classification déjà établies dans le contexte du méta-apprentissage et de les adapter à notre problème d'extraction d'événements.

En utilisant des techniques de méta-apprentissage, nous cherchons à extraire des caractéristiques pertinentes et discriminantes à partir des exemples annotés disponibles qui permettront de généraliser efficacement à de nouveaux types d'événements. Aujourd'hui, l'extraction de ces caractéristiques est facilitée par le développement des modèles de langue produisant des représentations de mots ou de phrases appelées plongements (*embeddings*). Ces modèles de langue sont généralement pré-entraînés sur de

grandes quantités de données textuelles permettant de capturer automatiquement des informations sémantiques et syntaxiques pour apprendre des représentations riches du langage.

L'exploitation de ces modèles de langue a considérablement renforcé les performances des modèles sur diverses tâches du TAL. Ces améliorations sont particulièrement notables dans le domaine de l'extraction d'information. Néanmoins, la tâche d'extraction d'événements reste difficile, surtout lorsqu'elle est abordée dans un contexte d'apprentissage à partir de peu d'exemples. La variabilité des descriptions et des structures linguistiques associées aux événements rend l'apprentissage et la généralisation plus complexes que pour d'autres tâches.

Nous adressons ces problématiques en présentant des techniques spécialement conçues pour l'extraction d'événements. Cela inclut l'utilisation de mécanismes d'attention pour extraire des représentations contextuelles pertinentes, l'exploitation efficace d'un modèle de langue pour obtenir des plongements plus riches en combinant ses différentes couches, un algorithme d'apprentissage contrastif mieux adapté pour la comparaison de représentations, ainsi que l'intégration des relations syntaxiques entre les entités et les déclencheurs d'événement pour améliorer les représentations des arguments.

Les axes de recherche de cette thèse peuvent être regroupés en trois catégories de questions scientifiques auxquelles nous tenterons de répondre.

Formulation du problème : il s'agit des questions portant sur la formulation et le cadre d'évaluation des sous-tâches d'extraction d'événements. En particulier, nous nous demandons :

- Comment pouvons-nous adapter les techniques de méta-apprentissage pour l'extraction d'événements à partir de peu d'exemples ?
- Quelles sont les meilleures stratégies pour aborder le problème de la classe *NULLE*[3] dans l'extraction d'événements à partir de peu d'exemples ?

Apprentissage de représentations : Cette catégorie concerne les questions portant sur la qualité des représentations fournies par les modèles de langue.

- Comment pouvons-nous améliorer la qualité de ces représentations pour l'extraction d'événements à partir de peu d'exemples ? Nous nous concentrerons en particulier sur les représentations du modèle BERT.

Algorithmes d'apprentissage et de classification :

3. En extraction d'information, la classe *NULLE* fait référence aux exemples qui ne sont affiliés à aucune des classes d'intérêt. Dans le contexte de l'extraction d'événements, cela englobe les termes qui ne sont pas des déclencheurs d'événements, mais qui doivent néanmoins être identifiés comme tels.

- Quels algorithmes de méta-apprentissage sont les plus efficaces pour l'affinage de modèles de représentation dans le contexte de l'extraction d'événements à partir de peu d'exemples ?
- Comment pouvons-nous évaluer de manière fiable la performance des modèles d'extraction d'événements à partir de peu d'exemples en utilisant des techniques de méta-apprentissage ?

1.3 . Contributions de la thèse

Les travaux de cette thèse se sont concentrés, dans un premier temps, sur la tâche de détection d'événements, puis sur la tâche d'extraction des arguments dans un second temps.

La première contribution a consisté à établir un cadre d'évaluation et à explorer les algorithmes de méta-apprentissage visant à transférer rapidement et efficacement les connaissances acquises sur des types d'événements vers de nouveaux types disposant de peu de données annotées. Pour ce faire, nous avons utilisé des réseaux prototypiques (Snell et al., 2017) en raison de leur simplicité et de leur efficacité démontrée dans d'autres tâches. Dans ce contexte, nous avons exploré des techniques visant à obtenir des représentations textuelles plus riches. En particulier, nous avons proposé une méthode consistant à combiner les couches cachées du modèle de langue BERT (Devlin et al., 2019a) afin d'exploiter au mieux les informations disponibles dans ce modèle. Cette première contribution a donné lieu aux publications (Tuo et al., 2022b) à la conférence NLDB-2022 [4], et (Tuo et al., 2022a) à la conférence TALN-2022 [5].

La deuxième contribution s'est concentrée sur la résolution d'un problème spécifique lié à l'extraction d'événements dans le contexte des réseaux prototypiques. Nous avons abordé la problématique de la classe des mots qui ne sont pas des déclencheurs, appelée classe *NULLE*, en proposant un algorithme d'apprentissage contrastif inspiré de la détection d'exemples hors domaine (*Out-of-Domain Detection*) ou détection d'anomalies. De plus, nous avons développé une méthode de seuillage dynamique pour décider si un mot est déclencheur ou non. Cette contribution a donné lieu à la publication (Tuo et al., 2023b) à la conférence ECIR-2023 [6] et la publication (Tuo et al., 2023a) à la conférence TALN-2023 [7].

Enfin, notre troisième contribution s'est focalisée sur l'extraction des arguments d'événements, en utilisant toujours des méthodes prototypiques. Dans cette section, nous avons évalué l'efficacité des méthodes développées dans les contributions précédentes pour cette tâche spécifique. De plus, nous avons démontré que l'enrichissement des

4. NLBD-2022
5. TALN-2022
6. ECIR-2023
7. TALN-2023

représentations des arguments par des informations syntaxiques, en complément des représentations fournies par les modèles de langue, pouvait conduire à des améliorations significatives de la performance des modèles.

En résumé, notre travail a introduit une méthode d'extraction d'événements à partir de peu d'exemples par transfert. Dans ce cadre, nous avons exploré diverses approches visant à enrichir les représentations fournies par un modèle de langue et proposé une nouvelle méthode pour traiter la classe *NULLE* dans ce contexte. Nos expérimentations ont démontré l'efficacité de ces méthodes dans les deux sous-tâches d'extraction d'événements, avec la perspective de pouvoir être étendues à d'autres tâches d'extraction d'information.

1.4 . Plan du manuscrit

Dans le chapitre 2, nous proposons une revue de la littérature sur l'extraction d'événements par apprentissage profond, en mettant un accent particulier sur les méthodes d'extraction à partir de peu d'exemples. Nous passons en revue les différentes formulations du problème, les modèles et les méthodes d'apprentissage utilisées dans la littérature. Cette revue nous permettra de situer notre travail par rapport aux avancées existantes et d'identifier les lacunes et les opportunités de recherche dans ce domaine.

Les chapitres 3 et 4 sont consacrés exclusivement à la détection d'événements à partir de peu d'exemples, en utilisant des réseaux prototypiques correspondant aux deux premières contributions de la thèse.

Dans le chapitre 5, nous présentons la troisième contribution sur l'extraction des arguments, formulée comme une tâche de classification de relations à partir de peu d'exemples.

Enfin, dans le chapitre 6, nous dressons un bilan de ce travail et discutons des limites de cette thèse et des perspectives pour les travaux futurs dans ce domaine.

Chapitre 2

État de l'art

Sommaire

Dans ce chapitre, nous proposons une revue de la littérature sur l'extraction d'événements avec un accent mis sur les méthodes d'extraction à partir de peu d'exemples. En nous appuyant sur la littérature existante, nous classifions les méthodologies d'extraction d'événements en plusieurs catégories selon les critères suivants :

- Le paradigme d'extraction : opposant les méthodes séquentielles ou jointes.

- La modélisation du problème : opposant les approches par classification, par génération ou par compréhension du langage (*Machine Reading Comprehension*, MRC).
- Les types de modèles : l'extraction par des règles, les modèles par apprentissage statistique et les modèles neuronaux.
- Le niveau de supervision : les approches supervisées, faiblement supervisées ou non supervisées. Mais aussi les hypothèses à priori sur les données en entrée (la connaissance des déclencheurs ou des entités, les bases de connaissances, les modèles pré-entraînés, etc.).

Nous examinons dans un premier temps l'état de l'art sur l'extraction d'événements supervisée. Puis, nous nous pencherons spécifiquement sur l'extraction d'événements à partir de peu d'exemples, qui est la problématique principale de cette thèse. Nous discuterons des travaux existants qui abordent ce défi et nous analyserons les différentes stratégies et méthodes qui ont été proposées pour surmonter les limitations dues au manque de données annotées. Nous présenterons d'abord des techniques mises en œuvre pour l'extraction d'événement à partir de peu d'exemples de façon générale. Puis, nous traiterons plus en détail la détection d'événements, qui vise à identifier la présence d'événements, et l'extraction des arguments, qui cherche à identifier les rôles des participants des événements ainsi identifiés.

Cette revue, en plus de synthétiser la littérature existante sur l'extraction d'événements à partir de peu d'exemples, permettra de soulever les défis liés à cette problématique, dont certains que nous traiterons dans cette thèse.

2.1 . L'extraction d'événements

2.1.1 . Définition de l'extraction d'événements

L'extraction d'événements est un axe de recherche de l'extraction d'information visant à extraire les structures d'événements dans des textes écrits en langage naturel. Elle consiste à identifier et extraire des informations spécifiques concernant des événements mentionnés dans des données textuelles. Étant donné que l'extraction d'événements est spécifique au domaine d'application, il n'existe actuellement aucune définition stricte d'un événement. Néanmoins, les campagnes d'évaluation au cours des années ont tenté chacune d'en donner des définitions de plus en plus précises.

Ces campagnes d'évaluation classent les événements en différents types, dont le nombre et le niveau de granularité peuvent varier. Chaque type d'événement est décrit de manière précise par des entités du texte appelées *Arguments* de l'événement. Bien que l'objectif principal de l'extraction d'événements soit d'identifier les arguments d'un événement et leur rôle vis-à-vis de lui, le jeu de données de la campagne ACE-2005 (Grishman et al., 2005; Walker et al., 2006) a introduit le concept de déclencheur événementiel

(*trigger* ou *event trigger*), désignant le mot ou le groupe de mots indiquant le plus explicitement possible la présence d'un événement dans une phrase. Le rôle principal de ce déclencheur est de créer un ancrage lexical afin d'aider à la recherche des arguments par la suite. L'extraction d'événements peut dès lors être envisagée en deux étapes : d'abord, l'identification des déclencheurs et leur classification en fonction d'un ensemble de types d'événements prédéfinis, tâche appelée détection d'événements (*Event Detection* ou *Trigger Detection*, ED) ; puis l'identification des arguments et la classification de ces arguments en fonction de leur rôle vis-à-vis de l'événement, appelée extraction des arguments (*Event Argument Extraction*, EAE).

2.1.2 . Campagnes d'évaluation et jeux de données

Les campagnes MUC (*Message Understanding Conference*). Les campagnes d'évaluation MUC sont une série de conférences organisées par le NRaD (*Naval Research and Development*) et cofinancées par la DARPA [1] (*Defense Advanced Research Projects Agency*) afin de stimuler la recherche en extraction d'information, dont l'extraction d'événements. MUC-1 (1987) était principalement une phase exploratoire où chaque groupe développait sa propre approche pour extraire les informations des documents, sans aucune évaluation formelle. La tâche a ensuite évolué vers le remplissage de formulaires (comme à la figure 1.1), avec des champs prédéfinis, lors de MUC-2 (1989). Cela a conduit à l'introduction de métriques d'évaluation telles que le rappel et la précision. Les éditions suivantes ont ensuite étendu l'ontologie à plus de types d'événements et complexifié petit à petit les champs à remplir dans les formulaires, allant jusqu'à 47 champs et 11 formulaires différents dans la cinquième édition (Grishman and Sundheim, 1996). Les deux dernières éditions (MUC 6 et 7) ont ensuite subdivisé la tâche en plusieurs sous-tâches plus ou plus moins indépendantes dans le but de rendre l'extraction d'événements plus modulaire.

Les campagnes ACE (*Automatic Content Extraction*). À la suite des campagnes MUC, les campagnes ACE [2] ont introduit la tâche de détection d'événements, introduisant par conséquent la notion de déclencheur événementiel, et l'extraction de relations, en plus de la tâche d'extraction des arguments d'événement. L'édition de l'année 2005 a marqué un tournant avec l'introduction du corpus ACE-2005, composé de 599 documents provenant de diverses sources, telles que les dépêches d'agence de presse, les bulletins télévisés, les blogs, les groupes de discussion en ligne et les transcriptions d'échanges téléphoniques. Cette diversité de sources a établi ACE-2005 comme un corpus de référence pour la tâche d'extraction d'événements. Le corpus ACE 2005, disponible en trois langues (anglais, mandarin et arabe), est composé de 8 types d'événement subdivisés en 33 sous-types plus fins et 34 rôles pour les arguments.

1. https ://www.darpa.mil
2. ACE-2005

Les campagnes TAC (*Text Analysis Conference*). Au cours des années 2010, les campagnes TAC ont succédé aux campagnes ACE, avec un objectif principal axé sur l'enrichissement des corpus et le peuplement de bases de connaissances. Ce programme a permis l'émergence de nouveaux corpus plus riches en types d'événements tels que TAC-KBP 2016 et 2017, ainsi qu'un nouveau standard d'annotation appelé ERE (*Entities, Relations, Events*). Le nouveau standard d'annotation ERE est largement similaire à celui d'ACE-2005, mais il a été simplifié afin de faciliter et harmoniser le processus d'annotation (Song et al., 2015). Comme dans le corpus ACE-2005, chaque mention d'événement est annotée en identifiant un déclencheur, en attribuant un type d'événement et en identifiant les entités participant à l'événement ainsi que leurs rôles. Cependant, certains types d'entités et de mentions (d'entités et d'événements) complexes ne sont pas pris en compte dans ce format. De plus, ERE introduit la notion de coréférence entre événements, ce qui permet de capturer les relations entre différentes mentions d'événements (Aguilar et al., 2014). L'évolution des campagnes TAC a donné lieu au corpus Rich-ERE en 2015 (Song et al., 2015).

Autres jeux de données En plus des jeux de données issus des campagnes précédentes, de nouveaux jeux de données ont récemment été introduits. Par exemple, le jeu de données MAVEN (*Massive General Domain Event Detection Dataset*) (Wang et al., 2020b), spécialement conçu pour la détection d'événements. Contrairement à ACE-2005 et Rich-ERE qui annotent les entités, MAVEN annote uniquement les déclencheurs d'événements, ce qui permet de passer de 38 types d'événements dans Rich-ERE à 168 types. Une version Maven-ERE (Wang et al., 2022) a été publiée récemment prenant en compte les relations de causalité, de temporalité et d'inclusion entre les événements, en suivant les propositions de l'approche « *Richer Event Description* » de O'Gorman et al. (2016).

FewEvent est un autre jeu de données conçu pour améliorer la couverture des types d'événements et servir de jeu d'évaluation pour la détection d'événements à partir de peu d'exemples (Deng et al., 2020; Lai et al., 2021a; Cong et al., 2021). En plus des types d'événements de ACE-2005 et TAC-KBP-2017, FewEvent intègre également des types d'événements supplémentaires provenant de sources telles que Freebase (Bollacker et al., 2008) et Wikipédia (Milne and Witten, 2008) extraits automatiquement par la méthode d'annotation automatique proposée par Chen et al. (2017).

Nous donnons dans l'annexe A plus de détails et des statistiques sur les jeux de données ACE-2005, MAVEN et FewEvent qui ont servi dans les expérimentations de la thèse.

2.1.3 . Modélisations de l'extraction d'événements

Comparaison entre les approches jointes et les approches séquentielles

L'extraction d'événements est souvent traitée suivant deux paradigmes : **les approches séquentielles** et **les approches jointes**. Les premiers systèmes d'extraction d'événements étaient principalement des approches séquentielles, qui abordaient le pro-

blème étape par étape, en identifiant d'abord les entités, puis les événements et enfin les arguments associés. Cependant, avec l'avènement des techniques d'apprentissage automatique, ces approches séquentielles ont été progressivement remplacées par des méthodes jointes plus sophistiquées (Riedel et al., 2009; Poon and Vanderwende, 2010; Riedel and McCallum, 2011; Li et al., 2013; Venugopal et al., 2014). Ces méthodes jointes visent à résoudre l'ensemble du problème d'extraction d'événements simultanément, en intégrant les différentes tâches dans un modèle global. Cette formulation globale tire parti de la capacité des modèles d'apprentissage à capturer les interactions complexes entre les différentes composantes d'un même processus d'extraction, ce qui conduit à de meilleures performances et à une prise en compte plus efficace des interactions entre différentes tâches. Toutefois, malgré les avantages liés aux méthodes jointes en termes de performances et de réduction de la propagation des erreurs inhérente aux approches séquentielles, elles présentent également certaines limitations. Étant donné que ces méthodes se concentrent souvent sur l'identification automatique des caractéristiques, il est parfois difficile de comprendre les raisons sous-jacentes des décisions prises par les modèles. Cette opacité peut être problématique dans les domaines dans lesquels il est important que les modèles soient interprétables, tels que le secteur médical ou juridique, où des explications claires sont nécessaires pour justifier les prédictions. Enfin, elles peuvent être sensibles à la qualité des données d'entraînement et à la représentativité des échantillons. En effet, la modélisation simultanée de toutes les tâches peut entraîner une augmentation de la complexité des modèles et une augmentation des temps de calcul. Cela peut poser des défis en termes d'efficacité, notamment lorsqu'il s'agit d'apprendre dans des environnements à ressources limitées.

L'un des premiers travaux à adopter cette modélisation jointe est le système *Beam-Joint* de Li et al. (2013), qui propose un modèle pour extraire simultanément les déclencheurs et les arguments. Les auteurs traitent ce problème à l'aide d'un modèle de prédiction structurée appris sur des caractéristiques lexicales et syntaxiques. Pour effectuer l'extraction, ils utilisent un décodeur de recherche par faisceau (*Beam Search*) qui permet de décoder simultanément la mention du déclencheur et les arguments. Une méthode similaire a été adopté par Araki and Mitamura (2015), qui résout simultanément la tâche de détection d'événements et de coréférence d'événements, ou encore (Yang and Mitchell, 2016; Zhang et al., 2019a; Nguyen and Nguyen, 2018a), qui traitent simultanément la détection d'événements et la reconnaissance des entités nommées.

Plus récemment, une nouvelle catégorie d'approches jointes a émergé, qu'on peut qualifier d'approches multitâches. Ces méthodes permettent d'entraîner des modèles sur plusieurs tâches simultanément, dans l'espoir que des tâches annexes puissent contribuer à améliorer la performance des tâches principales. Contrairement aux approches jointes traditionnelles, ces méthodes ne nécessitent pas toujours d'interaction directe

entre les différentes tâches, mais plutôt une agrégation des fonctions de coût de chaque tâche. De plus, l'inférence est souvent réalisée de manière séquentielle ou séparée, permettant une plus grande flexibilité dans l'utilisation de ces modèles.

Les techniques d'extraction

À l'origine, les systèmes d'extraction d'information étaient construits à partir de règles écrites à la main (Freitag and McCallum, 1999; Freitag and Kushmerick, 2000; Califf and Mooney, 1997, 2003). Ces méthodes avaient l'avantage d'être interprétables et offraient un contrôle précis sur le fonctionnement du système. Cependant, elles étaient coûteuses à développer et à entretenir, et leur couverture était limitée aux cas spécifiquement prévus par les règles. Ces méthodes à base de règles ont peu à peu laissé la place à des méthodes par apprentissage qui visent soit à apprendre à inférer automatiquement ces règles (Venugopal et al., 2014; Poon and Vanderwende, 2010; Riedel et al., 2009; Riedel and McCallum, 2011), soit à apprendre à extraire automatiquement des descripteurs (*features*) utiles à la résolution de la tâche, grâce aux méthodes par apprentissage profond (Chen et al., 2015a; Nguyen et al., 2016; Sha et al., 2018; Liu et al., 2018b). Ce sont ces méthodes qui prédominent aujourd'hui à cause de leur plus grande flexibilité et de leurs performances supérieures par rapport aux méthodes précédentes. L'évolution vers ces méthodes par apprentissage automatique a été motivée par la nécessité de traiter des tâches d'extraction d'informations plus complexes sur des données plus variées. Toutefois, ces méthodes peuvent manquer d'interprétabilité, en particulier les modèles d'apprentissage profond, ce qui peut rendre leur prise de décision opaque. De plus, elles nécessitent souvent de grandes quantités de données d'entraînement annotées pour obtenir de bonnes performances, ce qui peut être un obstacle dans les domaines dans lesquels les données sont rares. Nous allons concentrer la suite de cette revue sur ces méthodes par apprentissage profond, qui constituent l'état de l'art aujourd'hui.

2.2 . Extraction d'événements supervisée par apprentissage profond

Avec le développement des réseaux de neurones, de nouvelles méthodes d'extraction par apprentissage profond ont vu le jour. Cela inclut des méthodes à base de réseaux de neurones convolutifs (*Convolutional Neural Networks*, CNN) (Chen et al., 2015a; Nguyen et al., 2016; Björne and Salakoski, 2018), de réseaux de neurones récurrents (*Recurrent Neural Networks*, RNN) (Sha et al., 2018), de réseaux de convolution sur des graphes (*Graph Convolutional Networks*, GCN) (Liu et al., 2018b; Cui et al., 2020) et plus récemment, des architectures de modèles d'auto-attention (*Transformers*) (Vaswani et al., 2017). Nous donnons plus de détail sur ces réseaux à l'annexe B.

À partir de l'extraction automatique des descripteurs, différentes approches sont utilisées pour extraire les informations souhaitées. Initialement, les premiers travaux

considéraient le problème comme une **tâche de classification**. Toutefois, avec l'avènement des modèles génératifs, de nouvelles propositions ont émergé, traitant l'extraction d'événements comme une tâche de « **compréhension du langage** » (*Machine Reading Comprehension*, MRC) ou de **génération**.

2.2.1 . Approches par classification

Les premiers modèles d'extraction d'événements par apprentissage profond considéraient généralement l'extraction d'événements comme une tâche d'annotation de séquences (Chen et al., 2015b). Dans cette formulation, le texte est traité comme une séquence de mots auxquels on va associer des étiquettes pour marquer le type d'événement ou le rôle associé au mot en question.

(Nguyen and Grishman, 2015) est l'un des premiers travaux à avoir adopté une architecture CNN pour la détection d'événements. Dans ce modèle, chaque mot est initialement converti en une représentation vectorielle dense, obtenue en concaténant un plongement statique du mot, un plongement de position et un plongement des types des entités. Cette représentation vectorielle est ensuite utilisée comme entrée pour le réseau composé d'une couche de convolution, d'une couche d'agrégation *Max-Pooling* et d'un perceptron multicouche suivi d'une couche *softmax* pour classifier le mot.

Plusieurs améliorations ont ensuite été proposées à ce premier travail afin de l'adapter au mieux à la tâche d'extraction d'événements. Les CNN traditionnels extraient les caractéristiques les plus importantes en utilisant une seule valeur maximale pour chaque carte de caractéristiques via l'opération *Max-pooling*. Chen et al. (2015a) proposent une approche de « *Multi-pooling* » (*Dynamic Multi-pooling CNN*, DMCNN), où chaque phrase est divisée en plusieurs parties en s'appuyant soit sur les déclencheurs d'événements identifiés, soit sur les entités. Le module d'agrégation *Max-Pooling* est ensuite appliqué séparément sur chacune de ces parties de la phrase. Cette opération permet d'obtenir des représentations plus riches des mots et de mieux prendre en compte leur contexte local. Li et al. (2020b) reprennent la même idée pour proposer PMCNN (*Parallel Multi-Pooling CNN*) pour l'extraction d'événements dans le domaine biomédical. Nguyen and Grishman (2016) proposent une version *Non-consecutive CNN* (NC-CNN), qui permet de prendre en compte des fenêtres constituées de mots non consécutifs. L'objectif de ce modèle est de pouvoir capter un contexte plus lointain que ce que peut faire un CNN traditionnel. Dans cette même optique, Kodelja et al. (2019) proposent d'intégrer une représentation plus globale des contextes au-delà de la phrase.

Cependant, les CNN ne peuvent modéliser que les dépendances à courte distance à l'intérieur de la fenêtre définie par les filtres de convolution, limitant de ce fait la portée des interactions entre les éléments d'une séquence donnée. À l'inverse, les réseaux de neurones récurrents sont plus adaptés à la modélisation des interactions plus longues au sein des passages et moins sensibles à la position des mots. L'idée principale des

réseaux récurrents est de construire de façon récursive les représentations de chaque mot en prenant compte la représentation des mots précédents. Les réseaux bidirectionnels, introduits par Schuster and Paliwal (1997) en 1997, permettent en plus de capturer à la fois le contexte avant et après chaque mot en lisant la séquence dans les deux sens pour obtenir des informations contextuelles plus riches. Ces réseaux récurrents bidirectionnels ont également été adoptés pour la tâche d'extraction d'événements (Nguyen et al., 2016; Ghaeini et al., 2016; Chen et al., 2016) et donnent, en général, des performances plus élevées que les méthodes fondées sur les CNN. L'un des travaux de référence utilisant cette architecture est le système DyGIE (Luan et al., 2019) qui est une approche jointe exploitant les interactions entre les entités et les événements à partir de graphes de coréférences d'entités et les relations entre elles.

Plus récemment, de nombreux autres travaux ont adopté les modèles de type *transformer* depuis leur introduction par Vaswani et al. (2017). La distinction majeure entre ces modèles et les RNN réside dans le fait qu'ils ne nécessitent pas de traiter les données dans un ordre spécifique. Cette caractéristique confère un avantage majeur aux *transformers* par rapport aux RNN traditionnels, à savoir la possibilité d'effectuer une parallélisation plus efficace des calculs, se traduisant par des temps d'entraînement réduits ou des modèles de plus grande taille. L'une des premières propositions dans cette direction est DyGIE++ (Wadden et al., 2019), une version améliorée de DyGIE, qui a remplacé l'architecture BiLSTM par un encodeur BERT (Devlin et al., 2019a) tout en préservant l'architecture centrale de DyGIE. Le système oneIE (Lin et al., 2020) est une extension de DyGIE++ qui incorpore des caractéristiques globales issues de contraintes inter-tâches et inter-instances et une couche CRF pour la classification.

Les modèles présentés précédemment cherchent à extraire des descripteurs en exploitant les informations contextuelles dans les séquences de mots. Toutefois, une occurrence d'événement peut également être représentée sous la forme d'un graphe dont les nœuds correspondent au déclencheur et aux arguments de l'événement tandis que les arêtes sont associées aux rôles de ces arguments du point de vue du type de l'événement. Par ailleurs, une phrase peut aussi être représentée selon une structure de graphe, par exemple à travers une analyse en dépendances syntaxiques ou sémantiques. Cette observation a conduit à l'exploration des réseaux de convolution sur graphes (Graph Convolutional Networks, GCN) (Kipf and Welling, 2017a) pour l'extraction d'événements.

Réseaux de convolution sur des graphes

Les premiers travaux à employer les GCN pour l'extraction d'événements sont ceux de Nguyen and Grishman (2018b) et JMEE (Liu et al., 2018b), qui s'appuient tous les deux sur les graphes de dépendances syntaxiques des phrases. Leur objectif principal est de

tirer parti des interactions syntaxiques entre les mentions d'entités et les déclencheurs afin d'améliorer l'extraction d'événements. Le modèle de Nguyen and Grishman (2018b) est quasiment équivalent à celui de Nguyen and Grishman (2015), la seule différence étant que ces derniers associent un LSTM à un GCN au lieu d'utiliser un CNN, avec en final un gain significatif (4 points de pourcentage sur la F1-mesure). Néanmoins, ces méthodes présentent une limitation majeure en termes de prise en compte du contexte lointain dans les graphes, vu leur nombre limité de couches de convolution. En effet, trop augmenter le nombre de couches aboutit à des représentations similaires pour tous les nœuds du graphe étant donné que les graphes de dépendances syntaxiques sont entièrement connectés et relativement petits (de la taille d'une phrase). Dans un tel graphe, si l'on augmente trop le nombre de couches de convolution, tous les nœuds seront vus comme appartenant à un même voisinage. Pour lever cette limitation, Yan et al. (2019) proposent MOGANED (*Multi-Order Graph Convolution and Aggregated Attention for Event Detection*), qui permet de prendre en compte les voisinages distants en construisant un sous-graphe pour chaque chemin possible entre deux nœuds donnés. Les sorties de ces sous-graphes sont ensuite agrégées à l'aide d'un mécanisme d'attention, permettant ainsi de capturer des informations contextuelles plus riches et d'améliorer la performance de la détection d'événements. Lai et al. (2020b) proposent une approche alternative en utilisant un mécanisme de porte (*gate mechanism*) (Ruiz et al., 2019), qui permet de contrôler l'influence de chaque couche de convolution sur chaque nœud.

D'autres améliorations ont été proposées pour prendre en compte la sémantique des étiquettes dans les modèles. En effet, Cui et al. (2020) montrent que les étiquettes « nsujb » (sujets nominaux), « dobj » (objets nominaux) et « nmod » (modificateurs monimaux) représentent $32{,}2\%$ des étiquettes de dépendances liées aux déclencheurs (sur 40 types de dépendances considérées). Ils proposent donc le modèle EE-GCN (*Edge-Enhanced Graph Convolution Networks*), qui prend en compte le type des arêtes dans les graphes pour améliorer les représentations des nœuds. Le modèle GTN-ED (Dutta et al., 2021) traite le même problème, mais propose une méthode plus dynamique utilisant les GTNs (*Graph Transformer Networks*) (Yun et al., 2019). Cette approche consiste à calculer une matrice d'adjacence comme combinaison convexe des matrices d'adjacence liées à chaque type de relation (ici les dépendances syntaxiques) dans le graphe via un mécanisme d'attention. Cela permet d'obtenir une matrice d'adjacence globale avec des valeurs réelles contrairement à la version standard qui ne contient que des valeurs binaires pour dire si oui ou non deux nœuds sont voisins.

Plus récemment, GraphIE (Nguyen et al., 2022) propose une méthode fondée sur les graphes de dépendances entre les différentes sous-tâches d'extraction d'information

permettant d'aborder toutes les tâches d'extraction d'information de façon jointe. Ce travail est lui-même inspiré de systèmes tels que DyGIE et OneIE qui exploitent les interactions entre la reconnaissance des entités nommées et l'extraction d'événements.

2.2.2 . Approches par MRC

Les modèles de compréhension du langage (*Machine Reading Comprehension*, MRC) exploitent les capacités des modèles de langue à « comprendre » le langage naturel et résoudre des tâches en traitement automatique des langues. Ces méthodes consistent à répondre à des questions sur un passage textuel donné. Dans sa thèse, Chen (2018) définit le MRC comme un problème d'apprentissage supervisé qui considère des instances (p, q, a), où p est un passage textuel, q une question concernant ce passage et a, la réponse à cette question. L'objectif de cette tâche est d'apprendre un prédicteur f_θ prenant en entrée le passage p et la question correspondante q, et qui renvoie la réponse $a = f_\theta(p, q)$.

L'article de Li et al. (2019b) fait partie des travaux pionniers ayant proposé cette approche pour l'extraction des entités et des relations binaires. Cela a inspiré plusieurs autres travaux pour l'extraction d'événements (Du and Cardie, 2020; Liu et al., 2020; Li et al., 2020a). Étant donné la nature même de l'extraction d'événements, cette approche semble naturellement indiquée pour modéliser la tâche car l'extraction peut être résolue en répondant naturellement à des questions telles que « quoi ? », « qui ? », « quand ? », « où ? », « comment ? ».

Récemment, deux innovations majeures ont marqué cette approche. Tout d'abord, Feng et al. (2022) ont introduit une amélioration en associant deux types de questions : l'un vise à déduire un rôle par rapport à une entité et l'autre, de manière symétrique, vise à identifier une entité vis-à-vis d'un rôle d'argument. Cette approche cherche à renforcer la confiance du modèle dans ses réponses en créant ainsi deux signaux d'entraînement complémentaires. Ensuite, He et al. (2023) ont développé une méthode cherchant à limiter la propagation des erreurs en définissant un cadre de questions/réponses à plusieurs tours. Une telle approche permet au modèle de prendre en compte les prédictions précédentes pour améliorer les prédictions courantes et exploiter les dépendances hiérarchiques entre les arguments. Concrètement, He et al. (2023) commencent par poser une question afin d'identifier le déclencheur événementiel puis traitent les arguments un par un en prenant en compte les réponses précédentes dans la question courante. Cette idée est inspirée de Dai et al. (2022), qui proposent un processus similaire, mais par génération.

L'inconvénient majeur de cette approche réside dans le fait que la formulation des questions peut considérablement influencer les réponses générées. Toutefois, comme signalé plus haut, la tâche d'extraction d'événements peut facilement se traduire sous forme de questions. Par ailleurs, cette méthode nécessite de traiter les événements type

par type, car les questions sont spécifiques de chaque type d'événement et tous les types n'ont pas les mêmes arguments.

2.2.3 . Approches par génération

L'extraction d'événements a récemment été modélisée comme une tâche de génération de séquences en utilisant des modèles génératifs tels que BART (Lewis et al., 2020), T5 (Raffel et al., 2020) ou GPT (Brown et al., 2020; Radford and Narasimhan, 2018).

Ces approches offrent une plus grande flexibilité et une meilleure capacité de transfert vers de nouveaux types d'événements, permettant ainsi de traiter l'extraction d'événements avec peu de données. Plusieurs approches ont été proposées, allant de la modélisation du problème comme une tâche de traduction du langage naturel vers des structures d'événements (Paolini et al., 2021), à la génération par invite (*Prompting*) (Hsu et al., 2022; Ma et al., 2022; Li et al., 2021) ou encore l'augmentation de données (Pouran Ben Veyseh et al., 2021).

PLMEE (Yang et al., 2019) est l'un des premiers travaux à utiliser ces modèles pour la tâche d'extraction d'événements, à la fois pour résoudre le problème dit du chevauchement de rôle (*role overlap problem*) et pour tester les capacités de ces modèles à générer des exemples d'apprentissage pour enrichir les jeux de données existants. En effet, certaines entités peuvent avoir plusieurs rôles au sein d'un même événement. Par exemple, dans la phrase « *The explosion* ***killed*** *the bomber and three shoppers* », *the bomber* est à la fois l'attaquant, mais fait également partie des victimes de l'attaque. Prendre en compte ce phénomène nécessiterait donc de faire de la classification multi-label afin de pouvoir associer plusieurs étiquettes à cette entité, ce qui n'est pas possible avec les approches directes de classification multi-classes. Dans leur article, les auteurs proposent une méthode dans laquelle la prédiction est réalisée rôle par rôle (en tenant compte du type de l'événement), où le modèle prédit l'empan de l'argument correspondant au rôle en cours. Ainsi, un même empan peut être prédit plusieurs fois pour correspondre à des rôles différents. Ils combinent ensuite cette étape avec la détection des déclencheurs pour réaliser l'extraction d'événements de manière jointe. Pouran Ben Veyseh et al. (2021) proposent également une méthode d'augmentation de données à l'aide du modèle GPT-2 (Radford and Narasimhan, 2018).

Une seconde catégorie de ces approches génératives emploie plutôt une formulation par *prompting*. Une vue d'ensemble de ces méthodes par *prompting* est donnée à la figure 2.1. En résumé, ces méthodes construisent une invite à l'aide du passage et d'un préfixe contenant des informations supplémentaires (en général, des définitions ou des instructions liées à la tâche), encodent cette invite à l'aide d'un encodeur, puis génèrent une sortie, qui peut être conditionnée ou non par un patron, à l'aide d'un décodeur.

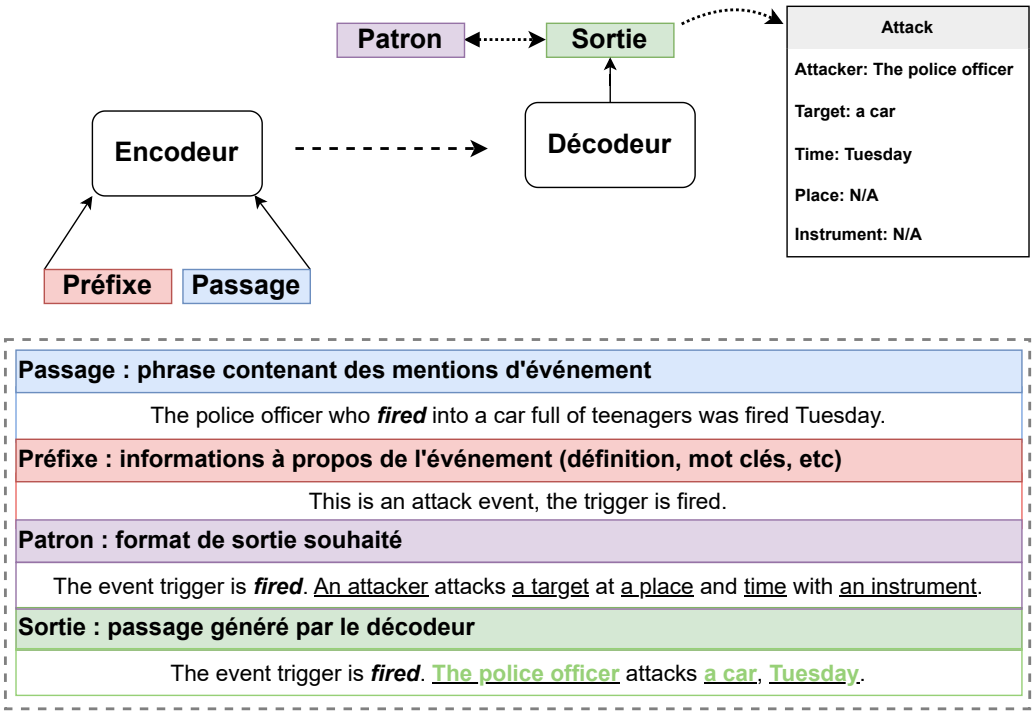


Figure 2.1 – Principe général des approches génératives par invite.

Les premières propositions dans cette catégorie de méthodes utilisaient des préfixes fixes à partir desquels elles construisaient l'invite (ou *prompt*) qu'elles concaténaient à la mention de l'événement pour servir d'entrée au modèle génératif. Cependant, les préfixes statiques ne tiennent pas compte des interactions entre les événements se trouvant dans un même contexte. Pour remédier à cette limitation, l'approche GTEE-DynPref (Liu et al., 2022b) remplace les préfixes statiques par des préfixes dynamiques, qui sont des vecteurs appris pendant l'entraînement. Cette combinaison de patrons d'événements manuels et de préfixes dynamiques permet de prendre en compte les interactions entre événements de façon implicite via un mécanisme d'attention entre les vecteurs du préfixe.

L'une des principales limitations de ces méthodes réside dans la qualité des données générées. Les phrases générées peuvent parfois être incohérentes ou peu naturelles, ce qui peut introduire du bruit dans l'ensemble des données générées et potentiellement nuire aux performances du modèle. De plus, les modèles de langue ont tendance à reproduire les biais présents dans les données d'entraînement, ce qui peut se traduire par une augmentation de ces biais dans les nouvelles données générées. Un autre défi est le contrôle sur le processus de génération. Les modèles génératifs peuvent avoir du mal à générer des exemples qui respectent des contraintes spécifiques, telles que des relations sémantiques complexes entre les événements et les entités. Il peut être difficile d'assurer que les exemples générés capturent correctement les caractéristiques souhaitées des nouvelles classes ou des nouvelles instances. Un autre problème concerne le

concept de « verbaliseur » (*verbaliser*) (Schick and Schütze, 2021a; Liu et al., 2023a). Un verbaliseur est une fonction qui lie le mot généré par le modèle génératif à l'une des classes à prédire. La recherche d'une telle fonction peut représenter un défi majeur, notamment lorsque les étiquettes sont complexes, comme ce peut être le cas pour les événements. Enfin, ces approches considèrent en général comme un pré-requis la connaissance du déclencheur d'événement et ne traitent pas les phrases ne contenant aucun événement. Cela les rend difficilement comparables avec les modèles qui ne font pas les mêmes hypothèses.

Bien que la philosophie générale entre les méthodes MRC et les approches génératives reste la même, ces deux approches diffèrent dans leur mise en œuvre et dans leurs objectifs spécifiques. Les méthodes génératives se concentrent sur la création de texte, tandis que les approches de MRC se concentrent sur l'extraction et la compréhension d'informations à partir de textes existants. Elles visent surtout à permettre à un système informatique de répondre à des questions ou de trouver des informations pertinentes dans un passage textuel.

2.2.4 . Comparaison des méthodes supervisées pour l'extraction d'événements

Dans le tableau 2.1, nous listons les articles abordant l'extraction d'événements par apprentissage sur les dix dernières années. Nous nous limitons aux articles donnant des résultats sur le jeu de données ACE-2005.

Performances des méthodes supervisées

Les métriques les plus fréquemment utilisées pour évaluer les performances des systèmes d'extraction d'information sont **le rappel**, **la précision** et **la F1-mesure**.

Le rappel évalue la capacité des systèmes à retrouver toutes les mentions pertinentes présentes dans les données. Il est calculé en divisant le nombre de mentions correctement extraites (vrais positifs) par le nombre total de mentions dans les données (toutes les étiquettes positives). Cela correspond au taux de vrais positifs trouvés parmi toutes les annotations positives.

$$R = \frac{\textit{Vrai Positif}}{\textit{Vrai Positif} + \textit{Faux Négatif}}$$

La précision évalue, de façon complémentaire, la capacité du système à extraire uniquement les mentions pertinentes et se calcule en divisant le nombre de mentions correctement extraites (vrais positifs) par le nombre total de mentions extraites par le système (ensemble des prédictions positives). Cela correspond au taux de vrais positifs

Année	Système	Méthode	Tâche			
			NER	REL	ED	EAE
2022	DEGREE (Hsu et al., 2022)	GEN	✓	✓	✓	✓
	GraphIE (Nguyen et al., 2022)	CLF	✓	✓	✓	✓
	PAIE (Ma et al., 2022)	GEN	✓		✓	✓
	GTEE-DynPref (Liu et al., 2022b)	GEN			✓	✓
2021	FourIE (Nguyen et al., 2021)	CLF	✓	✓	✓	✓
	AMRIE (Zhang and Ji, 2021)	CLF	✓	✓	✓	✓
	TANL (Paolini et al., 2021)	GEN	✓	✓	✓	✓
	TEXT2EVENT (Lu et al., 2021)	GEN		✓	✓	✓
	CLEVE (Wang et al., 2021c)	CLF	✓		✓	✓
	DualQA (Zhou et al., 2021)	MRC			✓	
2020	HPNet (Huang et al., 2020)	CLF	✓		✓	
	MQAEE (Li et al., 2020a)	MRC		✓	✓	
	BERT-QA (Du and Cardie, 2020)	MRC		✓	✓	
	RTM (Chen et al., 2020)	MRC				✓
	EEGCN (Cui et al., 2020)	CLF		✓		
	OneIE (Lin et al., 2020)	CLF	✓	✓	✓	✓
2019	GAIL-ELMo (Zhang et al., 2019b)	CLF		✓	✓	✓
	DyGIE++ (Wadden et al., 2019)	CLF	✓	✓	✓	✓
	HMEAE (Wang et al., 2019b)	CLF	✓			✓
	PLMEE (Yang et al., 2019)	GEN		✓	✓	✓
	JointTransition (Zhang et al., 2019a)	CLF	✓		✓	✓
	MLM-Joint (Li et al., 2019a)	CLF	✓		✓	✓
	Joint3EE (Nguyen and Nguyen, 2018b)	CLF	✓		✓	✓
2018	DEEB-RNN (Zhao et al., 2018)	CLF			✓	
	SELF (Hong et al., 2018)	GEN			✓	
	dbRNN (Sha et al., 2018)	CLF		✓	✓	✓
	JMEE (Liu et al., 2018b)	CLF	✓		✓	✓
2017	DMCNN-DS (Chen et al., 2017)	GEN	✓	✓		
	(Liu et al., 2017)	GEN			✓	
2016	JRNN (Nguyen et al., 2016)	CLF	✓		✓	✓
	JOINTEE (Yang and Mitchell, 2016)	CLF	✓		✓	✓
	BDLSTM-TNNs (Chen et al., 2016)	CLF	✓		✓	✓
	(Liu et al., 2016)	CLF			✓	
	NC-CNN (Nguyen and Grishman, 2016)	CLF			✓	
2015	DMCNN (Chen et al., 2015a)	CLF	✓		✓	
	(Nguyen and Grishman, 2015)	CLF		✓		
	(Araki and Mitamura, 2015)	CLF		✓		
2013	BeamJoint (Li et al., 2013)	CLF			✓	✓

Table 2.1 – Méthodes supervisées pour l'extraction d'événements en anglais sur les 10 dernières années ayant fourni des résultats sur le jeu de données ACE-2005. Nous comparons ces modèles en fonction de la formulation utilisée (CLF, pour classification, GEN pour génération et MRC pour Machine Reading Comprehension). Nous prenons également en compte l'aspect multitâche des modèles : NER pour la Reconnaissance d'entités nommées, REL pour l'extraction de relations, ED pour la détection d'événements et EAE pour l'extraction des arguments.

parmi toutes les prédictions positives.

$$P = \frac{\textit{Vrai Positif}}{\textit{Vrai Positif} + \textit{Faux Positif}}$$

La F1-mesure est une mesure combinant à la fois le rappel et la précision comme la moyenne harmonique de ces deux termes. Il donne une mesure globale de la performance du système en tenant compte à la fois des vrais positifs et des faux positifs. Plus sa valeur est élevée et se rapproche de 1, plus le modèle est performant. C'est cette mesure que nous allons principalement utiliser pour évaluer tous nos modèles.

$$F1 = 2 \times \frac{RP}{R + P}$$

À partir des tableaux 2.1 et 2.2, on peut noter que les approches par classification sont peu à peu remplacées par des approches par MRC et des approches génératives. Cela est dû aux avancées des modèles de langue actuels. Une autre observation importante est que les approches jointes et multitâches gagnent en popularité par rapport aux approches séquentielles. Ces approches permettent, en effet, une meilleure exploitation des interactions entre différentes tâches d'extraction d'informations, conduisant à une augmentation constante des performances au fil des années. Ces tendances reflètent les progrès constants dans le domaine de l'extraction d'événements, qui bénéficie de l'évolution rapide des techniques en apprentissage automatique et en traitement automatiques des langues.

2.3 . Extraction d'événements dans un contexte à faibles ressources

Les méthodes d'extraction d'événements supervisée, bien qu'ayant connu des avancées majeures, requièrent souvent des efforts humains importants pour l'obtention de données annotées pour leur entraînement. L'extraction d'événements à partir de peu d'exemples a donc suscité un intérêt croissant ces dernières années. Cette problématique est motivée par la nécessité de développer des systèmes d'extraction d'événements capables de s'adapter rapidement à de nouveaux scénarios et de nouveaux types d'événements, sans avoir à annoter de grandes quantités de données. Elle reste toutefois assez nouvelle et la tâche d'extraction d'événement est complexe. De ce fait, il manque encore un consensus sur les formulations du problème et les méthodes d'évaluation. Nous pouvons néanmoins classer les différentes propositions en trois catégories :

- Les approches à « faibles ressources » (*low resource*, LR), qui consistent à entraîner les modèles en utilisant seulement une petite partie de l'ensemble de don-

	Méthode	Déclencheurs			Arguments		
		P	R	F1	P	R	F1
GEN	DEGREE (Hsu et al., 2022)	-	-	73,3	-	-	55,8
	GTEE-DynPref (Liu et al., 2022b)	63,7	84,4	72,6	49,0	64,8	55,8
	TEXT2EVENT (Lu et al., 2021)	67,5	71,2	69,2	46,7	53,4	49,8
	PLMEE (Yang et al., 2019)†	81,0	80,4	80,7	62,3	54,2	58,0
	DMCNN-DS (Chen et al., 2017)†	75,5	66,0	70,5	62,8	50,1	55,7
	SELF (Hong et al., 2018)	71,3	74,7	73,0	-	-	-
	(Liu et al., 2017)	78,0	66,3	71,7	-	-	-
MRC	DualQA (Zhou et al., 2021)	-	-	-	49,1	42,3	45,4
	MQAEE (Li et al., 2020a)	-	-	73,8	-	-	55,0
	BERT-QA (Du and Cardie, 2020)	71,1	73,7	72,3	56,7	50,2	53,3
	RTM (Chen et al., 2020)	66,7	74,7	70,5	44,3	40,7	42,4
CLF	GraphIE (Nguyen et al., 2022)	-	-	74,8	-	-	60,2
	FourIE (Nguyen et al., 2021)	-	-	73,3	-	-	58,3
	CLEVE (Wang et al., 2021c)	78,1	81,5	79,8	55,4	68,0	61,1
	AMRIE (Zhang and Ji, 2021)	-	-	72,8	-	-	57,7
	OneIE (Lin et al., 2020)†	-	-	74,4	-	-	56,8
	HPNet (Huang et al., 2020)	80,1	75,7	77,8	64,6	50,7	56,8
	EEGCN (Cui et al., 2020)	76,7	78,6	77,6	-	-	-
	GAIL-ELMo (Zhang et al., 2019b)	74,8	69,4	72,0	61,6	45,7	52,4
	(Zhang et al., 2019b)	74,8	69,4	72,0	61,6	45,7	52,4
	DyGIE++ (Wadden et al., 2019)†	-	-	69,7	-	-	48,8
	HMEAE (Wang et al., 2019b)†	-	-	-	62,2	56,6	59,3
	JointTransition (Zhang et al., 2019a)†	74,4	73,2	73,8	55,7	51,1	53,3
	MLM-Joint (Li et al., 2019a)†	85,6	71,6	78,1	74,9	54,0	62,7
	Joint3EE (Nguyen and Nguyen, 2018b)	68,0	71,8	69,8	52,1	52,1	52,1
	dbRNN (Sha et al., 2018)	74,1	69,8	71,9	66,2	52,8	58,7
	JMEE (Liu et al., 2018b)	76,3	71,3	73,7	66,8	54,9	60,3
	ED-GCN (Nguyen and Grishman, 2018b)	77,9	68,8	73,1	-	-	-
	DEEB-RNN (Zhao et al., 2018)	72,3	75,8	74,0	-	-	-
	JRNN (Nguyen et al., 2016)	66,0	73,0	69,3	54,2	56,7	55,4
	JOINTEE (Yang and Mitchell, 2016)	75,1	63,3	68,7	70,6	36,9	48,4
	BDLSTM-TNNs (Chen et al., 2016)†	75,3	63,4	68,9	62,9	47,5	54,1
	(Liu et al., 2016)	77,6	65,2	70,7	-	-	-
	NC-CNN (Nguyen and Grishman, 2016)	-	-	71,3	-	-	-
	DMCNN (Chen et al., 2015a)	75,6	63,6	69,1	62,2	46,9	53,5
	(Nguyen and Grishman, 2015)†	-	-	69,0	-	-	-
	JointBeam (Li et al., 2013)	73,7	62,3	67,5	64,7	44,4	52,7

Table 2.2 – Comparaison des méthodes d'extraction d'événements supervisées sur le corpus ACE-2005 sur les dix dernières années. Tous ces résultats sont issus des papiers d'origine. †désigne les méthodes utilisant des ressources externes.

nées d'apprentissage, puis les évaluer sur un ensemble d'évaluation comportant les mêmes types d'événements. Ces méthodes permettent d'évaluer la capacité des modèles à généraliser à partir d'un nombre limité d'exemples (Ma et al., 2023c).

- Les approches par transfert (*class transfert*), qui consistent à entraîner les modèles sur un ensemble de classes, puis à les évaluer sur de nouvelles classes qui n'ont pas été vues pendant l'entraînement, avec peu de données annotées pour ces nouvelles classes. Ces méthodes permettent d'évaluer la capacité des modèles à transférer leurs connaissances vers de nouveaux domaines.

- Enfin, l'apprentissage sans exemple (*zero-shot learning*), qui est un cas particulier de transfert où aucune donnée annotée n'est fournie pour les nouvelles tâches. Dans ce cas, on utilise des informations annexes sur les données, telles que les définitions des classes, des informations issues des guides d'annotation, des structures de données ou encore des ressources externes. Ces méthodes nécessitent souvent l'utilisation de modèles pré-entraînés afin de tirer parti des informations qu'ils contiennent. Le terme « zero-shot » renvoie au fait qu'il n'est pas nécessaire d'avoir des exemples annotés pour effectuer des prédictions sur de nouvelles classes. L'un des premiers travaux d'apprentissage sans exemple en TAL est celui de Chang et al. (2008) et montre que le simple fait d'utiliser les noms des classes et des ressources externes permet d'obtenir des performances prometteuses sur des tâches de classification textes sur les jeux de données 20-Newsgroups (Lang, 1995) [3] et *Yahoo ! Answers* [4].

Néanmoins, les approches « *low ressource* » et « *class transfert* » peuvent être conciliées en combinant leurs principes dans un cadre d'apprentissage mixte, comme proposé par Ma et al. (2023c) pour la tâche de détection d'événements. Leur approche consiste à utiliser un modèle pré-entraîné comme « *backbone* » pour extraire des représentations vectorielles du texte. Ces représentations sont ensuite utilisées pour entraîner un classifieur spécifique aux nouvelles classes avec peu de données annotées. Cela permet de bénéficier à la fois de la capacité de généralisation des modèles pré-entraînés et de l'adaptation spécifique aux nouvelles classes grâce au classifieur.

Qu'elle que soit la formulation adoptée, les méthodes proposées ont abordé la tâche selon trois paradigmes : l'**augmentation de données**, le **transfert** de connaissances ou l'exploitation de **modèles de langue pré-entraînés**. Nous explorons plus en profondeur ces paradigmes dans les sous-sections à suivre, et ensuite, dans les sections 2.4 et 2.5, nous examinons séparément leur utilisation dans les sous-tâches de détection d'événements et d'extraction des arguments à partir de peu d'exemples.

2.3.1 . Augmentation de données

La façon la plus intuitive de pallier le manque de données d'apprentissage est d'augmenter les ensembles d'entraînement en y ajoutant de nouvelles données. Différentes

3. 20-Newsgroups
4. Yahoo! Answers

stratégies ont été utilisées pour augmenter les données, telles que l'utilisation de **ressources externes** comme des bases de connaissances ou des corpus non annotés, ou encore la **génération d'exemples synthétiques** à l'aide de modèles génératifs.

Les bases de connaissances Parmi les approches utilisant les bases de connaissances, (Liu et al., 2016) est le premier travail à exploiter FrameNet (Baker, 2014) pour la détection d'événements. En effet, un champ (*frame*) dans FrameNet est composé d'une unité lexicale et d'un ensemble d'éléments qui jouent respectivement des rôles similaires aux déclencheurs et aux arguments des événements du jeu de données ACE-2005. En plus de présenter des structures similaires, de nombreux champs dans FrameNet expriment en réalité certains types d'événements. Ces observations ont amené les chercheurs à explorer l'existence d'une correspondance entre ces champs et les types d'événements ainsi que la possibilité d'améliorer l'extraction d'événements en s'appuyant sur FrameNet. Cette idée a ensuite été étendue à l'extraction des arguments par Huang et al. (2016), qui utilisent en plus une analyse sémantique AMR (*Abstract Meaning Représentation*, AMR) pour obtenir des arguments candidats. Chen et al. (2017) et Zeng et al. (2018) proposent une approche similaire en utilisant Freebase et Wikipédia en plus de FrameNet. Par ailleurs, Ferguson et al. (2018) proposent une approche par *bootstrapping* partant du principe que différentes mentions d'un même événement dans des articles de presse doivent probablement être similaires. Ainsi, si un modèle d'extraction d'événements détecte certaines mentions d'événements avec une assez grande confiance au sein d'un ensemble d'articles, le modèle peut alors acquérir des exemples d'entraînement supplémentaires et diversifiés en ajoutant d'autres mentions similaires au sein du même ensemble d'articles. Cette hypothèse avait déjà été introduite par d'autres travaux d'extraction d'information auparavant, même si elle peut paraître un peu trop optimiste.

Les approches génératives Une autre méthode d'augmentation de données consiste à générer de nouvelles données à l'aide de modèles de langue. Dans cette famille d'approches, Yang et al. (2019) ont généré des données à partir du corpus ACE-2005 en trois étapes. Tout d'abord, les arguments dans une phrase sont remplacés par des arguments similaires pour créer de nouveaux exemples bruités. Ensuite, un modèle de langue est utilisé pour régénérer la phrase à partir de la phrase bruitée générée afin de créer de nouvelles phrases plus fluides. Enfin, les phrases candidates sont classées en utilisant le score de perplexité pour sélectionner les meilleures phrases ainsi générées. Plus récemment, Gao et al. (2022) ont proposé une méthode similaire, « *Mask-then-Fill* », qui consiste à remplacer des segments de phrases de longueurs variables à l'aide de modèles génératifs. Ils commencent par masquer de manière aléatoire un fragment de phrase, puis remplissent cet espace avec un modèle entraîné spécialement pour cette tâche de remplissage. L'avantage principal de leur méthode réside dans le fait qu'elle permet de rem-

placer un fragment de longueur arbitraire par un autre fragment de longueur variable, contrairement aux méthodes de « mots masqués » qui ne peuvent remplacer qu'un fragment de longueur fixe.

2.3.2 . Approches par transfert

Transfert par méta-apprentissage

L'objectif de l'apprentissage profond consiste à apprendre une fonction $f_\theta(x)$ paramétrée par les poids du modèle θ. De façon générale, pour apprendre ces paramètres θ, on introduit une fonction de coût $\mathcal{L}(\theta, \mathcal{D})$, où $\mathcal{D}$ représente l'ensemble de données utilisé pour l'entraînement. Cette fonction de coût est ensuite minimisée par un algorithme d'optimisation (par exemple, la descente de gradient) pour obtenir les paramètres optimaux $\theta^* = \arg\min_\theta \mathcal{L}(\theta, \mathcal{D})$.

Dans le cadre du méta-apprentissage, on cherche plutôt à déterminer un algorithme d'apprentissage permettant de généraliser rapidement sur de nouvelles tâches. Nous pouvons voir cela comme l'apprentissage d'une fonction $F_\phi(.)$ paramétrée par les méta-paramètres ϕ dont l'objectif est de trouver les paramètres optimaux θ^*. Cette approche est souvent décrite comme le fait d'*apprendre à apprendre* (Schaul and Schmidhuber, 2010). En effet, nous cherchons à apprendre les paramètres θ via l'apprentissage des méta-paramètres ϕ. Les méta-paramètres ϕ peuvent par exemple représenter l'architecture du modèle, l'initialisation des paramètres θ, le taux d'apprentissage, le choix de l'algorithme d'optimisation, etc. Ils sont en réalité des hyperparamètres propres au méta-apprentissage.

L'apprentissage de ces méta-paramètres ϕ se fait via des méta-tâches d'entraînement $\mathcal{T}_{train} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_N\}$, où les tâches $\mathcal{T}_i$ peuvent être issues du même problème ou non.

Chaque tâche $\mathcal{T}_i$ est définie par un ensemble de support (*support set*) $\mathcal{S}_i$ et un ensemble query (*query set*) $\mathcal{Q}_i$ constitués de données $\mathbf{x} = \{x_1, x_2, \ldots, x_K\}$ alignées sur leurs annotations $\{y_1, y_2, \ldots, y_K\}$, comme dans un problème d'apprentissage standard. Une tâche $\mathcal{T}_i$ peut être considérée comme un problème d'apprentissage standard ayant pour paramètres θ^i, où le support set joue le rôle d'ensemble d'apprentissage tandis que le query set peut être vu comme l'ensemble d'évaluation. On peut donc définir une fonction de coût sur les paramètres ϕ : $L(\phi, \mathcal{T}_{train}) = g\left(\mathcal{L}(\theta^i, \mathcal{T}_i)\right)$, où $g(.)$ est une fonction d'agrégation.

Les méta-paramètres pouvant être appris comme pour un problème d'apprentissage standard en parcourant les tâches $\mathcal{T}_i$ de $\mathcal{T}_{train}$. Une itération de mise à jour des paramètres θ par ce mécanisme est appelée épisode (*episode*).

Ce nouveau paradigme d'apprentissage permet d'améliorer la capacité d'adaptation d'un modèle à de nouvelles tâches et ainsi de réduire les efforts d'annotation spécifiques à chaque tâche.

Le méta-apprentissage a largement été utilisé ces dernières années pour l'extraction d'événements, en particulier pour la tâche de détection d'événements (Deng et al., 2020; Lai et al., 2021a; Cong et al., 2021; Tuo et al., 2022b, 2023b) et pour l'extraction des arguments, mais à l'échelle du document (Yang et al., 2023a). Étant donné que les contributions de cette thèse reposent sur ces approches, nous allons en donner plus de détails dans les chapitres à venir.

Transfert par distillation de connaissances

La distillation de connaissances est une technique d'apprentissage visant à transférer les connaissances d'un modèle complexe (appelé modèle enseignant, *teacher*) vers un modèle plus simple ou plus petit (appelé modèle étudiant, *student*) (Hinton et al., 2015). Le processus de distillation consiste à mettre à jour les poids du modèle étudiant en minimisant une fonction de coût qui tient à la fois compte des prédictions du modèle étudiant et de celles du modèle enseignant.

$$\mathcal{L}_{\mathsf{D}} = \mathcal{L}_{\mathsf{S}}(\hat{y}_{\mathsf{S}}, y) + \alpha \mathcal{L}_{\mathsf{T}}(\hat{y}_{\mathsf{T}}, y_{\mathsf{T}})$$

Dans cette famille d'approches, Liu et al. (2019) ont proposé un modèle de détection d'événements fondé sur l'apprentissage antagoniste (*adversarial learning*). Dans leur approche, le modèle enseignant est d'abord conçu pour apprendre les représentations des mots à partir des données annotées. Puis, un modèle étudiant est entraîné uniquement à partir de phrases non annotées en entrée mais contraint de produire les mêmes représentations vectorielles que le modèle enseignant sous la supervision d'un discriminateur antagoniste. De cette manière, le processus de distillation de connaissances à partir de phrases brutes est implicitement intégré à l'étape d'encodage du modèle étudiant. Cette approche permet de transférer les annotations du modèle enseignant vers le modèle étudiant. Ainsi, le modèle étudiant peut être directement utilisé pour la détection d'événements à partir de phrases non annotées.

Tong et al. (2020) ont également proposé une méthode de détection d'événements par distillation de connaissances à partir de WordNet (Fellbaum, 1998). Ils commencent par sélectionner des phrases de WordNet contenant des mots similaires aux déclencheurs pour augmenter les données d'entraînement. Mais, contrairement aux méthodes d'augmentation précédemment évoquées, leur approche de sélection ne dépend pas d'informations spécifiques aux événements, ce qui permet d'assurer une couverture plus large. Mais cela implique également l'utilisation de données bruitées qui ne sont pas directement liées aux événements. Ensuite, en utilisant les données enrichies, ils entraînent un modèle enseignant sur la tâche de détection d'événements. Ils entraînent en parallèle un modèle étudiant pour imiter les prédictions du modèle enseignant à partir des données non enrichies, en utilisant la divergence de Kullback-Leibler. L'idée sous-jacente est que le modèle enseignant apprend à partir des données bruitées, ce

qui lui confère une capacité de généralisation relativement faible dans le cadre des événements. En revanche, le modèle étudiant imite les prédictions du modèle enseignant en s'appuyant sur des données annotées du corpus, ce qui lui permet d'être plus précis. Cette approche permet non seulement de maintenir la distribution des données lors de l'inférence mais également de prédire les déclencheurs qui n'ont pas été vus pendant l'entraînement, ce qui est souvent le cas dans des contextes à ressources limitées. Dans une démarche similaire, Pouran Ben Veyseh et al. (2021) ont utilisé GPT-2 (Radford and Narasimhan, 2018) comme modèle enseignant. Mais, contrairement à Tong et al. (2020), leur objectif est de générer des données de haute qualité en exploitant les capacités de GPT-2 afin d'enrichir les données d'entraînement pour le modèle étudiant. Pour éviter que les bruits présents dans les données générées ne nuisent au processus d'entraînement, ils ont proposé d'entraîner le modèle enseignant en utilisant des connaissances ancrées dans les données originelles. Pour cela, ils ont poursuivi le pré-entraînement de GPT-2 sur la tâche de complétion de phrases de manière auto-régressive, tout en ajoutant des balises autour des déclencheurs. Cela permet à GPT-2 de générer de nouvelles données similaires aux données d'origine et annotées grâce aux balises. Ensuite, le modèle étudiant est entraîné sur une combinaison de données originales et de données générées, tout en étant guidé par les connaissances provenant du modèle enseignant grâce à une technique de transport optimal entre la distribution des données générées et celle des données originales. Cette approche permet de bénéficier des capacités de génération de GPT-2 tout en préservant les connaissances provenant des données originelles dans les cas où les données générées seraient trop bruitées.

2.3.3 . Exploitation des modèles de langue pré-entraînés

De façon générale, l'utilisation des modèles de langue pré-entraînés peut être vue comme une façon de transférer des connaissances des tâches de pré-entraînement vers des tâches cibles spécifiques, telles que l'extraction d'événements. Ces modèles peuvent même être parfois considérés comme des bases de connaissances spécifiques au domaine sur lequel ils ont été pré-entraînés (Petroni et al., 2019). Ils sont, en général, pré-entraînés sur de larges corpus textuels pour apprendre les structures d'une langue et sont capables de produire des représentations sémantiques riches pour diverses tâches du TAL. Cela explique la prééminence de ces modèles par rapport aux méthodes plus traditionnelles depuis la sortie de modèles pré-entrainés tels que ELMo (Peters et al., 2018) ou BERT (Devlin et al., 2019a), et en particulier dans un contexte à faibles ressources. Ces modèles de langue pré-entraînés peuvent, en plus, être affinés sur des jeux de données spécifiques d'extraction d'événements pour améliorer les performances et obtenir des représentations plus spécialisées. Par exemple, l'article « *Language Models are Few-Shot Learners* » de Brown et al. (2020) a démontré que les modèles de langue pré-entraînés peuvent être très efficaces pour s'adapter à de nouvelles tâches avec seulement quel-

ques exemples d'entraînement et sans aucun affinage. Dans ce travail, les auteurs ont notamment montré que le modèle GPT-3 était particulièrement performant sur une bonne dizaine de tâches du TAL sans affinage supplémentaire. Ils n'avaient pas évalué ses capacités sur les tâches d'extraction d'informations, mais les évaluations dans ce sens se sont multiplié depuis la sortie de InstructGPT (Ouyang et al., 2022) [5] en novembre 2022 (Wei et al., 2023; Ma et al., 2023b; Li et al., 2023).

Cependant, l'utilisation de ces modèles de langue pré-entraînés présente également des défis. Ces modèles sont souvent très massifs et nécessitent des ressources de calcul importantes pour leur pré-entraînement et leur affinage. De plus, ils peuvent être limités par les biais et les erreurs présents dans leurs données de pré-entraînement. Néanmoins, l'utilisation de modèles de langue pré-entraînés offre un cadre prometteur pour l'extraction d'événements, permettant aux modèles d'apprendre à généraliser à partir de peu d'exemples.

2.4 . Détection d'événements à partir de peu d'exemples

Les travaux ayant abordé l'extraction d'événements à partir de peu d'exemples se sont principalement concentrés sur la tâche de détection d'événements (*Few-Shot Event Detection*, FSED) en raison de sa relative simplicité par rapport à l'extraction des arguments.

Ces travaux peuvent être subdivisés en plusieurs catégories, chacune ayant des objectifs spécifiques. Une première ligne de recherche vise l'**identification d'événements**, qui consiste à identifier les déclencheurs événementiels étant donnés les types d'événements. Le premier travail dans ce contexte est celui de Bronstein et al. (2015), qui utilise des listes de mots clés associés à chaque type d'événement afin de calculer des descripteurs sémantiques entre ces mots clés et les mots exemples à classifier. L'objectif ici est d'exploiter les liens entre les déclencheurs et ces mots clés pour distinguer les déclencheurs des autres mots. Étant donné que ces mots clés sont des exemples de déclencheurs [6], ces derniers ont des liens particuliers, le même lemme ou relation de synonymie par exemple, avec les déclencheurs au sein des exemples. Cette approche a été enrichie et améliorée par Lai and Nguyen (2019) pour mieux discriminer les déclencheurs en utilisant des réseaux de neurones convolutifs combinés à un mécanisme d'attention impliquant ces mots-clés. En outre, Chen et al. (2021b) proposent un modèle ajoutant une prise en compte explicite des interactions entre les déclencheurs et leurs contextes pour résoudre le problème dit de la malédiction des déclencheurs (*trigger curse problem*). En effet, il existe un fort déséquilibre entre les déclencheurs pour un événement donné. Une grande majorité des événements est marquée par très peu de déclencheurs diffé-

5. InstructGPT est un modèle qui ajoute l'apprentissage par instructions au-dessus d'un modèle GPT-3.
6. Les mots clés ici sont les exemples de déclencheurs donnés dans le guide d'annotation.

rents, conduisant de ce fait à un sur-apprentissage de ces déclencheurs. Une meilleure prise en compte du contexte, sans la mention du déclencheur, doit donc permettre de réduire ce phénomène. Leur méthode permet de trouver le bon équilibre entre la prise en compte du contexte et la représentation des déclencheurs.

Une deuxième ligne de recherche s'est intéressée à la **classification d'événements**, qui vise à trouver les types d'événements à partir de la connaissance des déclencheurs. Cette sous-tâche de classification a été explorée par plusieurs études qui ont utilisé principalement des méthodes de méta-apprentissage (Snell et al., 2017; Vinyals et al., 2016; Sung et al., 2018; Geng et al., 2019). Ces travaux ont en particulier été dynamisés par la publication de l'article de Deng et al. (2020), associé au jeu de données FewEvent et utilisant les réseaux prototypiques (Snell et al., 2017).

Enfin, la **détection d'événements** a été abordée dans une troisième ligne de recherche, où les travaux visent à combiner l'identification d'événements et la classification d'événements (Cong et al., 2021; Tuo et al., 2022b; Zhang et al., 2022c; Yu et al., 2022). Ces travaux peuvent être classifiés en deux grandes familles d'approches : **les approches prototypiques** et **les approches génératives** (*prompting*).

2.4.1 . Les approches prototypiques

Depuis la publication de Deng et al. (2020), plusieurs autres travaux ont suivi dans la perspective du recours aux réseaux prototypiques. Deng et al. (2020) ont proposé un modèle appelé *Dynamic-Memory-Based Prototypical Network*, qui s'appuie sur un réseau prototypique à « mémoire dynamique » pour apprendre de meilleurs prototypes pour chaque type d'événements. Contrairement à un simple moyennage comme proposé dans le réseau prototypique d'origine, les prototypes sont calculés de manière itérative grâce à un mécanisme d'attention entre les épisodes permettant la prise en compte d'interactions entre les différents épisodes. Deng et al. (2020) ont effectué leurs évaluations sur le jeu de données FewEvent fourni avec l'article établissant un nouvel état de l'art pour la tâche. Lai et al. (2021a) reprennent le même principe de garder en mémoire des traces des épisodes passés en introduisant la notion de prototypes inter-épisodes (*Prototype Across Task*). D'autres travaux ont proposé d'enrichir les prototypes à l'aide de connaissances extérieures. Le modèle KE-PN (Zhao et al., 2022) propose, par exemple, d'aligner automatiquement les types d'événements à la base de connaissances FrameNet, puis d'appliquer une méthode auto-supervisée pour filtrer les entrées bruitées venant de FrameNet. Leur modèle est également renforcé par un module de classification des relations entre types d'événements pour améliorer la performance globale. Zhang et al. (2022b) proposent une approche *zero-shot*, mais utilisent les définitions des événements comme prototype et apprennent à rapprocher les concepts de leurs définitions par apprentissage contrastif. Une première phase de pré-entraînement permet d'aligner les définitions des concepts de WordNet avec des phrases correspondantes

mentionnant ces concepts. La méthode se poursuit avec une phase d'affinage apprenant à aligner les définitions des événements avec des concepts similaires de WordNet. La dernière étape est une phase d'inférence où chaque phrase est classifiée en fonction de sa similarité avec les définitions des types d'événements, un seuil de similarité décidant si la phrase est une mention ou non.

Si ces travaux cherchent à construire de meilleurs prototypes pour chaque type d'événements, d'autres travaux se sont plutôt concentrés sur les mécanismes d'apprentissage. Par exemple, Lai et al. (2020a) introduisent des fonctions de coût spécifiques inspirées des scores de *clustering* (Silhouette) pour améliorer la séparabilité entre les classes tout en minimisant la dispersion au sein de la même classe. Deng et al. (2021) proposent quant à eux d'utiliser des ontologies d'événements pour aider la tâche de détection d'événements. Ils créent une ontologie d'événements fondée sur des corrélations entre événements (type, sous-type, relation temporelle ou de causalité) et proposent d'entraîner conjointement un encodeur permettant à la fois de prédire les déclencheurs d'événements et les relations entre événements. La partie consacrée à la détection d'événements est fondée sur un réseau prototypique standard et les relations entre événements sont considérées comme une tâche annexe pour aider à la détection.

Il existe par ailleurs des approches ressemblantes parues avant la publication de l'article sur les réseaux prototypiques en 2017. Par exemple, Peng et al. (2016) ont proposé une méthode consistant à générer et encoder des structures d'événements, puis classifier les mentions d'événements en fonction de leur similarité avec la représentation de chaque événement. Pour cela, ils utilisent les exemples d'événements issus du guide d'annotation (comme Bronstein et al. (2015)), appliquent du SRL sur ces quelques exemples, encodent les résultats obtenus et calculent la représentation de l'événement comme la moyenne des représentations des exemples.

2.4.2 . Les approches génératives et par MRC

Plusieurs travaux ont également abordé la tâche de détection d'événements avec des approches génératives dont nous avons déjà parlé. La plupart de ces travaux ont d'abord présenté des modèles dans le contexte supervisé et montré qu'ils pouvaient également fonctionner dans un contexte à faibles ressources grâce à la capacité de généralisation des modèles de langue. Nous allons présenter certains d'entre eux, tels que RTM (Du and Cardie, 2020), EERC (Liu et al., 2020), PAIE (Ma et al., 2022), DUAL-QA (Feng et al., 2022) ou encore DEGREE (Hsu et al., 2022) à la section 2.5 qui concerne à la fois les déclencheurs et les arguments.

PILED (Li et al., 2022b) fait partie des travaux ayant abordé uniquement la tâche de détection d'événements. Li et al. (2022b) y proposent une approche en deux étapes : une première étape de classification de phrases par *prompting* (Schick and Schütze, 2021a) ; puis une seconde étape où ils localisent la position du déclencheur au sein de la phrase.

Méthode	Entrée	Sortie
EEQA (Du and Cardie, 2020)	X. Q : What is the trigger in the event ?	R : **fired**
EDTE (Lyu et al., 2021)	Premise : X. Hypothesis : This text is about an Attack event. ... Premise : X. Hypothesis : This text is about a Start-Org event.	**Yes.** ... No.
PET (Schick and Schütze, 2021b)	X. The word *fired* triggers a `[MASK]` event. ... X. The word *car* triggers a `[MASK]` event.	**Attack** ... N.A.
UIE (Lu et al., 2022)	<spot> Attack <spot> Attack <spot> ... <spot>. X.	(Attack : fired)
DEGREE (Hsu et al., 2022)	X. *DESCRIPTION*(Attack). Event trigger is `[MASK]`. ... X. *DESCRIPTION*(Attack). Event trigger is `[MASK]`.	Event trigger is **fired** ... Event trigger is N.A.

Table 2.3 – Exemples d'entrée/sortie pour différentes méthodes par MRC et par génération. X est la mention d'événement : *"The police officer who fired into a car full of teenagers was fired Tuesday."*, dans lequel le mot *"fired"* est déclencheur d'un événement de type *"Attack"*.

Toutefois, la phase d'identification du type d'événement repose sur un *verbalizer* qui doit être bien construit pour que le système fonctionne correctement. Le verbalizer est une fonction qui établit une correspondance entre le mot prédit par le modèle génératif et les étiquettes des classes. ZEOP (Zhang et al., 2022d) propose une méthode sans verbalizer en considérant directement le plongement du mot masqué fourni par le modèle de langue. Plus récemment, Yue et al. (2023) ont introduit MetaEvent, qui utilise un algorithme de méta-apprentissage comme MAML (Finn et al., 2017b) et une phase d'inférence par complétion de phrases (*prompting*). Contrairement à la méthode de Zhang et al. (2022d), leur modèle combine les descripteurs des déclencheurs et la sortie de l'invite (*prompt*) pour prédire les types d'événements.

Si ces méthodes cherchent à s'affranchir du verbalizer, elles n'abordent pas spécifiquement les problèmes liés à l'invite en elle-même. Une étude récente (Wang et al., 2023a) propose « *The Art of Prompting* », qui compare plusieurs types d'invite et propose un cadre pour la construction d'invites pour la détection d'événements.

La détection d'événements à partir de peu d'exemples a également été abordée comme une tâche de compréhension du langage (MRC), allant des formulations sous forme de questions/réponses (Boros et al., 2022; Du and Cardie, 2020; Liu et al., 2020) aux méthodes par reconnaissance d'implication textuelle (*Recognizing Textual Entailment*, RTE) (Lyu et al., 2021; Sainz et al., 2022a). Boros et al. (2022) proposent d'exploiter les informations sur les entités pour améliorer la détection d'événements en marquant les entités par des balises spécifiques, comme Liu et al. (2016). Feng et al. (2020) combinent une approche RTE pour la détection des déclencheurs et un système de questions/réponses pour l'extraction des arguments.

Nous donnons des exemples d'entrée/sortie pour ces méthodes par MRC et par génération dans le tableau 2.3.

2.4.3 . Synthèse des différentes méthodes

Dans le tableau 2.4, nous présentons un comparatif des méthodes de détection d'événements à partir de peu d'exemples en mettant en évidence leurs différentes formulations et prérequis. Ces approches se distinguent par leur formulation du problème, la taille des encodeurs utilisés, le nombre d'exemples, l'utilisation de ressources externes (**Res. Ext**.) ainsi que la prise en compte des annotations des entités en entrée (**Ents.**). Dans ce tableau, k-shots signifie k exemples par classe et $p\%$ représente la proportion de l'ensemble d'entraînement utilisé.

	Méthode	Encodeur	#Exemples	Res. Ext.	Ents.
Proto	Seed (Bronstein et al., 2015)	-	30		
	MSEP (Peng et al., 2016)	-	172	✓	
	ZSL (Huang et al., 2018)	CNN	0	✓	
	DMBPN (Deng et al., 2020)	BERT	{5,10,15}-shots		
	OntoED (Deng et al., 2021)	BERT	{0,1,5,10,15,20}%		
	ILP-EE (Zhang et al., 2021a)	BERT_L	0	✓	✓
	PA-CRF (Cong et al., 2021)	BERT	{5,10}-shots		
	ProAcT (Lai et al., 2021a)	BERT	{5,10}-shot		
	CausalED (Chen et al., 2021b)	BERT	5-shot		
	Keyword (Yu et al., 2022)	RoBerta_L	176	✓	
	ZED (Zhang et al., 2022b)	BERT	0-shot	✓	
	HCL-TAT (Zhang et al., 2022c)	BERT	{5,10}-shot		
	KE-PN (Zhao et al., 2022)	BERT	{1,5}-shot	✓	
Prompt	Text2Event (Lu et al., 2021)	T5	{1,5,25}%		
	UIE (Lu et al., 2022)	T5	{1,5,10}%		
	DEGREE (Hsu et al., 2022)	BART_L	{1,5,10, 20}%		
	PILED (Li et al., 2022b)	BERT	{5,10}-shots		
MRC	FSQA (Feng et al., 2020)	BERT	{0,1,3,5,7,9}-shots		
	EERC (Liu et al., 2020)	BERT	{0,1,5,10,20}%		
	BERT-QA (Boros et al., 2022)	BERT	0-shot		✓
	EDTE (Lyu et al., 2021)	BERT	0-shot		

Table 2.4 – Comparaison des propositions pour la détection d'événements à partir de peu d'exemples.

Dans notre étude, les ressources externes utilisées dans ces approches peuvent englober des bases de connaissances ainsi que des informations issues de la résolution d'autres tâches proches de l'extraction d'événements, telles que l'AMR et le SRL. Cependant, nous ne considérons pas les guides d'annotation et les modèles de langue comme des ressources externes, bien qu'ils aient été pré-entraînés sur des données externes à la tâche d'extraction d'événements.

2.5 . Extraction des arguments à partir de peu d'exemples

Comme nous l'avons vu précédemment, l'extraction d'événements supervisée a connu des avancées significatives grâce aux méthodes d'apprentissage profond. Par consé-

quent, ce sont ces méthodes qui ont été les plus explorées pour l'extraction d'arguments à partir de peu d'exemples. Comme pour les méthodes supervisées, les propositions dans un contexte avec peu de données se répartissent en trois grands types de formulation : les formulations par classification, par MRC ou par génération.

De nombreux systèmes ont abordé cette tâche comme un problème de compréhension du langage, parmi lesquels Du and Cardie (2020) ont proposé d'entraîner deux modèles BERT en parallèle, l'un pour la détection des événements et l'autre pour l'extraction des arguments. Cette approche par question-réponse a été approfondie par Zhou et al. (2021), qui ont introduit une méthode à double question. Ils posent la question « quel est le rôle de l'entité ? » pour une entité donnée et la question « quelle entité joue ce rôle ? » pour un rôle donné. Cette approche à double question peut être considérée comme une approche multitâche où les deux tâches se complètent mutuellement.

Une autre méthode par génération appelée « *Reading The Manual* » a été introduite par Chen et al. (2020). Cette méthode permet l'extraction des arguments en remplissant des patrons spécifiques pour chaque type d'événement. L'idée générale est de remplir des « trous » spécifiques dans chaque patron en utilisant des entités extraites du texte (ou en les laissant vides si aucune entité ne correspond au rôle). Cette approche a été améliorée par Dai et al. (2022), qui remplissent les patrons de façon itérative et bidirectionnelle. Cela apporte deux avantages principaux par rapport au modèle d'origine : le caractère itératif du remplissage des patrons permet de tirer parti des prédictions précédentes pour affiner les prédictions courantes et sa bidirectionnalité permet de prendre en compte à la fois le contexte avant et après l'entité considérée. Ces approches par *prompting* ont été significativement améliorées par les systèmes DEGREE (Hsu et al., 2022) et PAIE (Ma et al., 2022) publiés presque simultanément en 2022. Les systèmes DEGREE et PAIE utilisent tous les deux le modèle pré-entraîné encodeur-decodeur BART (Lewis et al., 2020) pour aligner la sortie souhaitée sur le passage en entrée. Ces approches sont similaires aux méthodes de remplissage de patrons, mais sont mises en œuvre de manière plus automatisée grâce à la génération contrainte. PAIE prend en compte le déclencheur événementiel en entrée tandis que DEGREE prend en considération des listes de mots clés déclencheurs issues du guide d'annotation (comme Bronstein et al. (2015)). En ce qui concerne les prédictions des arguments, PAIE se concentre sur la prédiction des empans (*spans*) des arguments tandis que DEGREE prédit directement les arguments grâce au décodeur. L'approche de PAIE fondée sur les empans permet de limiter le phénomène d'hallucination (Maynez et al., 2020; Ji et al., 2023) observé dans les modèles génératifs [7] en s'assurant que les prédictions du modèle proviennent uniquement du passage considéré.

7. Le terme « hallucination » est apparu pour la première fois dans le domaine de la vision par ordinateur dans les travaux de Baker et Kanade (Baker and Kanade, 2000).

Enfin, Lyu et al. (2021) propose une méthode fondée sur la reconnaissance d'implication textuelle. À partir d'une entité du texte, ils construisent des tâches de RTE en utilisant le passage de l'événement comme prémisse et l'hypothèse selon laquelle l'entité joue un certain rôle. L'objectif de cette tâche est de prédire si l'hypothèse implique la prémisse. Sainz et al. (2022a) utilisent le même principe, mais introduisent en plus des contraintes de conformité entre les types des entités et les rôles qu'elles peuvent tenir dans les événements. En effet, le guide d'annotation de ACE-2005 indique que certains rôles ne peuvent être tenus que par certains types d'entités. Par exemple, dans un événement de type « Transport », l'origine et la destination ne peuvent être que des lieux. Ce filtrage par les types des entités a également été adopté par Lin et al. (2023).

Même si la grande majorité de ces méthodes n'a pas été spécifiquement conçue pour l'apprentissage à partir de peu d'exemples, les auteurs démontrent leur efficacité dans des scénarios avec peu d'exemples, en utilisant un faible pourcentage des données d'entraînement, voire sans exemples dans certains cas. Nous avons, par conséquent, déjà présenté certaines d'entre elles dans les sections précédentes.

Extraction des arguments sans exemple (*Zero-Shot Argument Extraction*) Lu and Roth (2012) ont été les pionniers dans le domaine de l'extraction d'événements sans exemple. Ils ont proposé de modéliser le problème en utilisant des champs aléatoires conditionnels semi-Markoviens (*Semi Markov Conditional Random Fields*). Leur modèle extrait conjointement les déclencheurs d'événements et les arguments d'événements à partir des structures d'événements et des mentions d'entités. Cette approche fondée sur les structures des événements a inspiré Huang et al. (2018), dont la méthode permet de s'affranchir des mentions d'événements et des entités en entrée. Pour cela, ils utilisent une analyse AMR (*Abstract Meaning Representation*, AMR) (Flanigan et al., 2014; Wang et al., 2015) pour générer des déclencheurs et des entités candidates. Ils utilisent ensuite un réseau CNN pour encoder les structures d'événements et les graphes générés par l'analyse AMR et classifient les déclencheurs en fonction des similarités entre les structures d'événement et les graphes AMR (comme Peng et al. (2016)). Ils extraient ensuite les arguments en fonction des similarités entre les nœuds des graphes AMR et les nœuds dans les structures d'événements. Plus récemment, Yu et al. (2022) ont proposé un modèle fondé également sur la similarité entre des plongements des paires (entité/déclencheur) avec la sémantique des rôles. Zhang et al. (2022f) adoptent une approche similaire par transfert de la tâche d'étiquetage des rôles sémantiques (*Semantic Role Labelling*, SRL) vers l'extraction des arguments d'événements en tirant parti des similarités entre leurs structures d'arguments. Cependant, cette méthode avait encore des limitations en termes de performance. Dans une étude ultérieure, Zhang et al. (2021a) proposent une approche similaire et l'utilisation de ressources externes (NYT (Sandhaus, 2008) [8]) pour enrichir les

8. Corpus NYT

données. Ils ajoutent ensuite une étape de filtrage de contraintes de conformité entre les types des entités et leurs rôles, qu'ils incorporent par programmation linéaire en nombres entiers (*Integer Linear Programmation*, ILP Roth and Yih (2004)). Toutes ces propositions confirment bien la théorie de Pustejovsky (1991) stipulant que les mentions d'événements peuvent être transposées aux structures d'événements de façon systématique : « *the semantics of an event structure can be generalized and mapped to event mention structures in a systematic and predictable way.* ». Pour un type donné, l'idée de ces travaux est de faire correspondre les mentions d'événements à la structure du type de l'événement. En d'autres termes, une mention peut être vue comme une occurrence particulière de la description générale de l'événement.

Dans une autre perspective, les méthodes de Lyu et al. (2021) et Sainz et al. (2022a) fondées sur le RTE sont également utilisées en zero-shot. En outre, Lin et al. (2023) proposent une approche similaire à celle de Sainz et al. (2022a), mais cette fois-ci par génération. Pour un argument candidat donné, les auteurs écrivent manuellement toutes les hypothèses, puis évaluent la cohérence entre chacune des hypothèses et le passage mentionnant l'événement à l'aide d'un modèle de langue GPT-J (Wang and Komatsuzaki, 2021). Le rôle correspondant à l'hypothèse ayant le score le plus élevé est sélectionné comme prédiction initiale, qui est ensuite filtrée à l'aide de contraintes issues du guide d'annotation. Plus récemment, Wang et al. (2023b) ont testé la capacité des gros modèles de langue (*Large Language Models*, LLM) sur la tâche d'extraction des arguments sans exemple par génération de code. À partir de la définition des types d'événement et des rôles, ils génèrent du code permettant d'identifier les arguments dans des passages contenant des événements.

Contrairement à la tâche de détection, l'extraction des arguments a été peu abordée par méta-apprentissage. La seule étude à proposer une telle approche est celle de Yang et al. (2023a) mais, elle est faite au niveau document et non pas au niveau phrastique. De plus, cette étude présente des performances très basses (de l'ordre de 10% de F1-mesure) par rapport aux méthodes supervisées. L'une des contributions de cette thèse propose justement une approche pour l'extraction des arguments par méta-apprentissage que nous aborderons dans le chapitre 5. Ce sera également l'occasion de comparer plus en détail les modèles d'extraction des arguments avec peu de données.

2.6 . Conclusion

Dans ce chapitre, nous avons examiné les différentes approches et méthodes utilisées pour l'extraction d'événements, partant des approches supervisées vers les méthodes d'extraction à partir de peu d'exemples. Nous avons présenté un aperçu des tâches et défis liés à l'extraction d'événements, en mettant en évidence les différents types de jeux de données et les campagnes d'évaluation les plus utilisés dans ce do-

maine. Nous avons notamment pu voir que la campagne ACE 2005 a été un tournant décisif pour l'extraction d'événements en introduisant le concept de déclencheur, qui ancre spécifiquement les événements dans le texte. Cette modélisation a eu pour conséquence de focaliser la tâche sur le niveau intra-phrastique, une caractéristique partagée par la plupart des approches actuelles, alors qu'elle avait initialement été conçue pour être effectuée au niveau document. Cette étude nous a permis de mettre en évidence plusieurs tendances majeures dans le domaine de l'extraction d'événements.

Tout d'abord, nous avons observé l'émergence des méthodes neuronales, qui ont largement supplanté les approches traditionnelles utilisant des traits définis manuellement. Les modèles neuronaux ont démontré leur efficacité en construisant de manière automatique une représentation des exemples et en apprenant des motifs complexes à partir des données d'entraînement.

Nous avons ensuite constaté une adoption croissante des modèles pré-entraînés, en particulier les modèles de langue tels que BERT (Devlin et al., 2019a), GPT (Brown et al., 2020), T5 (Raffel et al., 2020) ou encore BART (Lewis et al., 2020). Ces modèles, pré-entraînés sur de grands corpus de textes, ont acquis une importance grandissante dans plusieurs domaines du TAL grâce à leur capacité à apprendre des représentations linguistiques riches et généralisables, ce qui les rend très performants pour les tâches d'extraction d'événements, en particulier dans un contexte à faibles ressources. L'adoption de ces modèles de langue pré-entraînés a conduit à de nouvelles perspectives pour l'extraction d'événements. Auparavant, cette tâche était souvent considérée comme un problème d'annotation de séquences. Cependant, les modèles de langue pré-entraînés ont été construits sur la base de tâches de modélisation du langage, telles que la prédiction de mots masqués ou la prédiction de la phrase suivante. Cette propriété de ces modèles a incité les chercheurs à reformuler la tâche d'extraction d'événements en des tâches de compréhension du langage ou de génération de texte, exploitant ainsi pleinement les capacités de ces modèles pour traiter le texte de manière plus « naturelle ».

Nous avons par ailleurs identifié deux grandes tendances dans l'extraction d'événements à partir de peu d'exemples. D'une part, les approches par transfert, consistant à adapter des modèles pré-entraînés, ont été privilégiées pour leur simplicité. Ces méthodes permettent d'utiliser des connaissances préalablement acquises sur des corpus volumineux dans un domaine source pour améliorer les performances dans le domaine cible. Ces modèles ont surtout été adoptés pour la tâche de détection des déclencheurs. La seconde grande tendance est l'adoption des approches par génération, qui exploitent des modèles génératifs récents pour produire des descriptions d'événements directement à partir du contexte. Bien que plus complexes et diversifiées dans leur mise en œuvre, ces approches semblent plus efficaces pour traiter l'extraction des arguments. Ces deux approches offrent des perspectives intéressantes pour améliorer l'extraction d'événements dans des scénarios à faibles ressources, chacune ayant ses avantages et

ses inconvénients. Des études telles que (Ma et al., 2023c) ont également tenté de combiner les deux méthodes pour tirer parti des avantages de chacune d'entre elles.

Cette analyse révèle toutefois certaines limites. Tout d'abord, plusieurs modèles de l'état de l'art souffrent d'un problème de reproductibilité en raison de la complexité des prétraitements spécifiques à chacun de ces travaux de recherche. Deuxièmement, les cadres d'évaluation des méthodes d'extraction d'événements varient en fonction des jeux de données et des métriques utilisés, ce qui les rend peu comparables, surtout avec l'utilisation de modèles de langue pré-entraînés et dans les scénarios à faibles ressources. En outre, ces approches ne considèrent pas toujours les mêmes connaissances a priori, rendant leur comparaison encore plus discutable. Enfin, notre analyse s'est principalement concentrée sur les approches en domaine fermé, au niveau phrastique, laissant de nombreux défis à relever dans le contexte de l'extraction d'événements en domaine ouvert ou au niveau document.

Enfin, l'utilisation croissante de modèles de langue pré-entraînés soulève des questions éthiques et de confidentialité des données, exigeant une attention particulière dans leur développement. Cela vaut en particulier pour des modèles tels que ChatGPT [9]. Ces modèles massivement paramétrés ont démontré leur remarquable capacité à générer du texte cohérent et à accomplir une variété de tâches linguistiques. Bien que leur conception initiale ne soit pas spécifiquement orientée vers l'extraction d'information, ils ont été évalués sur des tâches d'extraction d'information et ont affiché des performances prometteuses sur certaines tâches, mais limitées dans l'ensemble (Li et al., 2023). Leur flexibilité et leur aptitude à appréhender les relations complexes dans le texte en font des pistes d'intérêt pour l'avenir de la recherche en extraction d'événements. L'article de Ma et al. (2023b) propose, par exemple, une approche novatrice qui combine ces modèles avec des modèles de langue plus petits. Leur méthode « *filter-then-rerank* » consiste à effectuer une première prédiction avec un petit modèle de langue, puis à soumettre les meilleures réponses de ce modèle, sous forme de questionnaire à choix multiples, à un gros modèle tel que ChatGPT. Cette approche hybride ouvre ainsi des perspectives prometteuses pour l'amélioration de l'extraction d'événements et souligne l'importance d'exploiter les synergies entre différents types de modèles. Enfin, Ma et al. (2023a) proposent l'approche STAR (*Structure-to-Text datA GeneRation*), qui génère des exemples d'événements annotés à partir de patrons d'événements. Leur approche est également très prometteuse pour l'augmentation de données.

9. ChatGPT

Chapitre 3

Enrichissement de l'encodeur pour la détection d'événements à partir de peu d'exemples

Sommaire

Dans ce chapitre, nous présentons la première contribution de la thèse visant à utiliser une approche par méta-apprentissage pour la détection d'événements à partir de peu d'exemples. Nous adoptons spécifiquement une méthode par méta-apprentissage reposant sur les réseaux prototypiques, qui demeurent prédominants pour cette tâche comme avons pu le voir dans la revue de la littérature. Nous explorons plusieurs stratégies permettant d'enrichir les représentations vectorielles des mots dans le cadre de cette tâche. En particulier, nous examinons des stratégies pour mieux exploiter les informations présentes dans les couches cachées d'un modèle de langue pré-entraîné et des méthodes d'enrichissement par injection de connaissances. Cette idée émane du principe que les couches des modèles de langue multicouches portent des informations de natures différentes, qu'il est donc intéressant d'exploiter afin d'enrichir la représentation des données à traiter. Il est donc intéressant de pouvoir extraire ces informations de façon explicite pour enrichir les ensembles de données.

3.1 . Introduction

Les méthodes traditionnelles d'extraction d'événements supervisée sont très coûteuses puisqu'elles nécessitent de grands corpus annotés manuellement. Un des défis actuels est donc de développer des méthodes permettant de réduire, dans la mesure du possible, le coût de développement de ces systèmes. C'est dans ce contexte que nous envisageons la détection d'événements, par le biais de l'apprentissage à partir de peu d'exemples (*Few-Shot learning, FSL*) (Lake et al., 2015).

Diverses configurations ont été explorées : la généralisation de modèles à de nouveaux types d'événements à l'aide de listes de mots clés (Bronstein et al., 2015; Lai and Nguyen, 2019; Yu et al., 2022), l'enrichissement des données avec des ressources externes (Deng et al., 2021), l'apprentissage sans exemples (Zhang et al., 2021b) et l'apprentissage à partir de peu d'exemples qui nous intéresse dans cette étude (Shen et al., 2021; Chen et al., 2021b; Cong et al., 2021).

Ces différentes contributions se sont surtout concentrées sur l'utilisation de réseaux prototypiques (Snell et al., 2017) pour résoudre la tâche. Dans ce chapitre, notre contribution se focalise sur une meilleure exploitation des représentations fournies par le modèle de langue BERT pour la détection d'événements à partir de peu d'exemples ; plus spécifiquement, en utilisant d'une part de l'injection de connaissances et, d'autre part, différentes façons d'associer les couches cachées du modèle pour obtenir de meilleures représentations.

Nos expériences montrent que non seulement, ces stratégies permettent d'améliorer significativement les performances sur la tâche, mais aussi qu'elles peuvent être cumulées avec d'autres types d'amélioration.

3.2 . Détection d'événements à partir de peu d'exemples par méta-apprentissage

Nous formulons le problème de la détection d'événements comme un problème d'annotation de séquences (Ramshaw and Marcus, 1995), au format BIO (*Beginning Inside Outside*), qui peut être ramené à une tâche de classification multi-classes au niveau des mots.

Le format BIO est une convention d'annotation couramment utilisée pour les tâches d'annotation de séquences. Ce format permet de représenter des annotations de séquences en indiquant le début (B pour *Beginning*) ainsi que les parties internes (I pour *Inside*) des déclencheurs et les mots qui ne font partie d'aucun déclencheur (O pour *Outside*). Nous donnons un exemple d'annotation BIO dans le tableau 3.1 ci-dessous.

X	The	police	officer	who	**fired**	into	a	car	full	of	teenagers	**was**	**fired**	Tuesday	.
Y	O	O	O	O	**B-Attack**	O	O	O	O	O	O	**B-End-Pos.**	**I-End-Pos.**	O	O

Table 3.1 – Exemple de données d'entrée : X représente une phrase découpée en mots et Y la séquence d'étiquettes associée. Nous n'avons annoté que les déclencheurs d'événements, mais les arguments peuvent également être annotés de la même façon. **End-Pos.** correspond à un événement de licenciement (*End-Position*) et **Attack** à une attaque conflictuelle.

Comme nous l'avons vu dans la revue de la littérature, la détection d'événements à partir de peu d'exemples a surtout été traitée par des méthodes de méta-apprentissage [1] depuis la publication de Deng et al. (2020) introduisant le jeu de données FewEvent.

L'utilisation du méta-apprentissage vise à pallier le manque de données annotées pour entraîner les modèles. Cette approche consiste à entraîner les modèles sur certaines tâches afin de les évaluer sur de nouvelles tâches similaires, mais qui n'auraient pas été observées lors de l'entraînement. Ainsi, le modèle « apprend à apprendre » à partir des tâches d'entraînement, ce qui lui permet de généraliser sur d'autres tâches lors de l'évaluation. Les tâches d'entraînement et d'inférence peuvent comporter différentes classes et même appartenir à des domaines différents. Dans ce contexte, les travaux utilisent généralement une formulation épisodique (Vinyals et al., 2016) dans laquelle le modèle apprend à partir d'épisodes correspondant à des scénarios d'apprentissage avec peu de données. L'idée sous-jacente est que si le modèle peut résoudre efficacement les tâches avec peu d'exemples pendant l'entraînement épisodique, il devrait également être performant lors de l'évaluation sur de nouvelles tâches.

1. En tout cas jusqu'en 2021, au moment de la réalisation de ces travaux.

3.2.1 . Apprentissage épisodique « *N-ways, k-shots* »

L'apprentissage épisodique N-ways, k-shots est un cadre de méta-apprentissage proposé par Vinyals et al. (2016) pour la classification d'images à partir de peu d'exemples. Ce cadre simule une situation dans laquelle on dispose d'un nombre limité d'exemples annotés pour chaque classe et effectue le méta-entraînement en itérant sur un grand nombre de scénarios avec diverses tâches. Chacun de ces scénarios est appelé épisode. Chaque épisode d'entraînement consiste en la création d'une tâche de classification, composée de N classes, chacune ayant k exemples.

Formellement, à chaque épisode, le modèle prend en considération un sous-ensemble de données étiquetées $\mathcal{S}$, appelé *Support set*, qui contient N types d'événements, et k exemples annotés par type (k étant généralement petit, par exemple compris entre 1 et 10) :

$$\mathcal{S} = \{(x_1^1, y^1), \ldots, (x_k^1, y^1), \ldots, (x_1^N, y^N), \ldots, (x_k^N, y^N)\}$$

où $x_i^n = \{w_1, \ldots, w_L\}$ est une séquence de longueur L contenant un déclencheur événementiel de la classe n et y^n est la séquence d'étiquettes associée. Nous disposons par ailleurs d'un autre sous-ensemble similaire $\mathcal{Q}$, appelé *Query set*, dans lequel les échantillons font l'objet d'une prédiction en s'appuyant sur l'observation des exemples du support set. Dans le cadre de l'apprentissage épisodique N-ways, k-shots, un épisode est défini comme $\mathcal{E} \triangleq \{\mathcal{S}, \mathcal{Q}\}$, où $\mathcal{S}$ représente le support set et $\mathcal{Q}$ le query set associé. L'entraînement dans ce contexte consiste à mettre à jour les poids d'un encodeur en fonction de la prédiction sur les éléments du query set.

Nous donnons un aperçu de cette méthode à la figure 3.1.

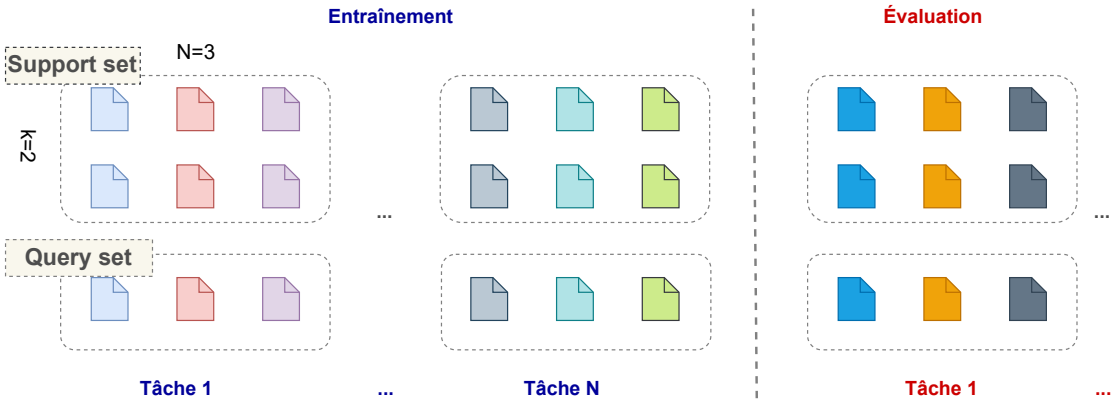


Figure 3.1 – Principe général de l'apprentissage épisodique N-ways, k-shots.

3.2.2 . Les algorithmes de méta-apprentissage fondés sur un apprentissage de similarité

Avant l'émergence des modèles génératifs, la détection d'événements était principalement abordée comme une tâche de classification de mots. Par conséquent, les re-

cherches pour la détection d'événements à partir de peu d'exemples se sont naturellement dirigées vers des algorithmes de classification à partir de peu d'exemples. Ces algorithmes reposent sur l'idée générale de créer une représentation pour chaque classe (appelée prototype) et de catégoriser les exemples en fonction de leurs similarités à ces prototypes. En raison de leur simplicité et de leur efficacité, ces algorithmes ont largement été utilisés dans le domaine de la vision par ordinateur pour la classification d'images à partir de peu d'exemples, ainsi que dans plusieurs tâches d'extraction d'information telles que la Reconnaissance d'Entités Nommées (Fritzler et al., 2018), l'Extraction de Relations (Gao et al., 2019b; Popovic and Färber, 2022) et l'Extraction d'Événements (Deng et al., 2020; Cong et al., 2021; Yang et al., 2023b). Différentes variantes de ces approches prototypiques ont été proposées, chacune apportant ses particularités. Nous présentons les trois principales ci-dessous.

Réseaux de correspondance, *Matching Networks* Vinyals et al. (2016) ont été les premiers à proposer un algorithme de méta-apprentissage fondé sur une métrique et à introduire le concept d'apprentissage épisodique par la même occasion. Leur algorithme était initialement conçu pour la classification d'images avec un seul exemple annoté par classe (*one-shot learning*). L'idée fondamentale derrière les Matching Networks est de créer une correspondance (*matching*) entre les points de l'ensemble de requêtes (*query set*) et les points de l'ensemble de support (*support set*). Pour classifier un point de requête, on commence par calculer ses distances avec tous les points de support. Ces distances sont ensuite utilisées pour attribuer un poids à chaque exemple de support, reflétant son importance pour classer le point de requête. Enfin, le point de requête est classifié en utilisant le vote pondéré des étiquettes des points de support, selon les poids attribués à chaque point. Les Matching Networks sont une version pondérée des k-plus proches voisins (kNN) dans laquelle on attribue une importance relative à chaque voisin. Contrairement aux kNN traditionnels, les Matching Networks attribuent un poids à chaque voisin à partir des exemples du support set.

Réseaux prototypiques, *Prototypical Networks* Les réseaux prototypiques, proposés par Snell et al. (2017), sont fondés sur l'idée d'apprendre une représentation pour chaque classe à partir de l'ensemble de support. Cette représentation, appelée prototype, est la moyenne des exemples de la classe dans l'ensemble de support. Les réseaux prototypiques peuvent être vus comme une version simplifiée des réseaux de correspondance avec deux différences majeures : (i) au lieu de comparer directement les représentations des exemples dans l'ensemble de support, ils utilisent les représentations des classes ; (ii) les auteurs évaluent l'importance du choix de la fonction de similarité et montrent que la distance euclidienne fonctionne mieux que la similarité cosinus pour la classification d'image à partir de peu d'exemples. Si ces deux réseaux diffèrent dans le cadre du few-shot, ils sont équivalents dans le scénario one-shot.

Réseaux de relation, *Relation Networks* Les réseaux de relation, introduits par Sung et al. (2018), se distinguent des deux réseaux précédents par leur approche pour calculer la fonction de similarité. De manière similaire aux réseaux prototypiques, le réseau de relation construit un prototype de chaque classe comme moyenne des exemples de la classe dans l'ensemble de support. Chaque prototype est ensuite concaténé au résultat de l'encodage de la requête et transmis à un module de relation pour calculer sa similarité avec les prototypes. Ce calcul est réalisé par un réseau de neurones, qui apprend la fonction de similarité pour chaque problème et permet ainsi de s'affranchir du choix a priori d'une telle fonction. En effet, l'efficacité des fonctions de similarité doit être évaluée en tenant compte du problème traité et de l'encodeur qui fournit les représentations vectorielles de chaque exemple. Les valeurs de similarité seront fortement influencées par les directions et magnitudes de ces représentations vectorielles. Plusieurs études ont d'ailleurs montré que la normalisation des vecteurs (Chen et al., 2019; Ji et al., 2020; Tian et al., 2020) ou leur répartition uniforme dans l'espace (Ding et al., 2021) pouvaient améliorer les performances des modèles par rapport à des méthodes sans aucune normalisation. Toutefois, même si le choix de la métrique de similarité n'est plus un problème, l'architecture du réseau de calcul des similarités devient un nouvel hyper-paramètre à prendre en compte lors de la conception des réseaux de relation.

Ces trois types de réseaux peuvent être comparés à des approches antérieures, telles que l'algorithme des k plus proches voisins (k-*Nearest Neighbors*, k-NN), dans leur processus de prédiction. En effet, les réseaux de correspondance sont assimilables à une version pondérée des kNN. De plus, ils partagent des similitudes avec les réseaux siamois (Koch et al., 2015) et les réseaux par triplet (*triplet networks*) (Hoffer and Ailon, 2015) dans leur principe d'apprentissage. Par la suite, nous appellerons l'ensemble de ces méthodes les approches prototypiques puisqu'elles reposent toutes sur la construction d'un prototype pour représenter chaque classe. Le cadre prototypique pour l'apprentissage épisodique est souvent décomposé en quatre étapes distinctes, chacune jouant un rôle particulier dans le processus global de classification. Les diverses approches développées dans ce domaine ont exploré des améliorations spécifiques pour chacun de ces modules, visant à optimiser les performances de l'extraction d'événements à partir d'un nombre limité d'exemples. Ces quatre modules, dont l'organisation est illustrée à la figure 3.2, sont les suivants :

Un module d'échantillonnage englobant toutes les stratégies relatives à la sélection d'échantillons, que ce soit pendant la phase d'entraînement des modèles ou lors de leur évaluation. Alors que la proposition initiale de Vinyals et al. (2016), qui a introduit le concept de N-ways, k-shots, envisageait un nombre fixe de classes et d'exemples par classe, les stratégies d'échantillonnage ont évolué pour s'adapter aux besoins spécifiques des tâches et aux caractéristiques des jeux de données. Certains travaux se sont

penchés sur l'utilisation de méthodes d'échantillonnage dynamiques permettant d'ajuster la fréquence d'apparition des classes et du nombre d'exemples par classe (Wang et al., 2021a; Bragg et al., 2021) afin de mieux refléter la réalité des jeux de données.

Un module d'encodage Le module d'encodage est un élément critique dans le processus d'apprentissage épisodique. Il se concentre sur la création de représentations vectorielles significatives pour chaque exemple ou chaque classe, ces représentations jouant un rôle majeur dans la généralisation du modèle à de nouvelles classes. En effet, si les représentations vectorielles sont suffisamment riches en informations discriminantes et capturent efficacement les caractéristiques distinctives de chaque classe, alors le modèle sera en mesure de bien généraliser à de nouvelles classes. Notre contribution dans ce chapitre ne porte que sur ce module.

Un module prototypique, qui concerne la manière de construire les prototypes à partir du *support set* ou de combiner les informations des prototypes ou des interactions entre eux.

Un module de prédiction, qui est dédié au calcul des similarités entre les prototypes et les exemples du *query set*, ainsi qu'à la manière dont les poids du modèle sont mis à jour lors de l'entraînement. Cela englobe des aspects tels que le choix de la fonction de similarité, qui mesure la proximité entre les exemples et les prototypes, et la sélection d'une fonction coût pertinente pour mettre à jour les poids de l'encodeur. L'objectif est de garantir que les prototypes et les exemples du *query set* soient correctement appariés, permettant ainsi au modèle de réaliser une classification précise malgré la faible quantité d'exemples disponibles. Ce module peut également couvrir l'ajout d'un réseau de classification supplémentaire ou d'une couche CRF permettant de capturer les interdépendances entre étiquettes.

De façon générale, ces méthodes prototypiques cherchent à construire un prototype pour chaque classe avec les exemples du support set et classifient les exemples du query set en fonction de leur similarité à ces prototypes. Formellement, cela se traduit par :

$$c^k(\mathcal{S}) = g(w_i | \{\forall w_i \in \mathcal{S}, y_i = k\}) \qquad P(y = k | w_q, \mathcal{S}) = \frac{\exp(s(f_\theta(w_q), c^k))}{\sum_{j=1}^{N} \exp(s(f_\theta(w_q), c^j))} \tag{3.1}$$

où $P()$ est une fonction d'activation *softmax* qui permet d'estimer la probabilité d'appartenance de l'exemple w_q à la classe k, c^k le prototype de la classe k, $f_\theta()$ représente l'encodeur paramétré par θ, $g()$ est une fonction d'agrégation, $s(.,.)$ une fonction de similarité qui peut être paramétrique ou non, et N le nombre total de classes.

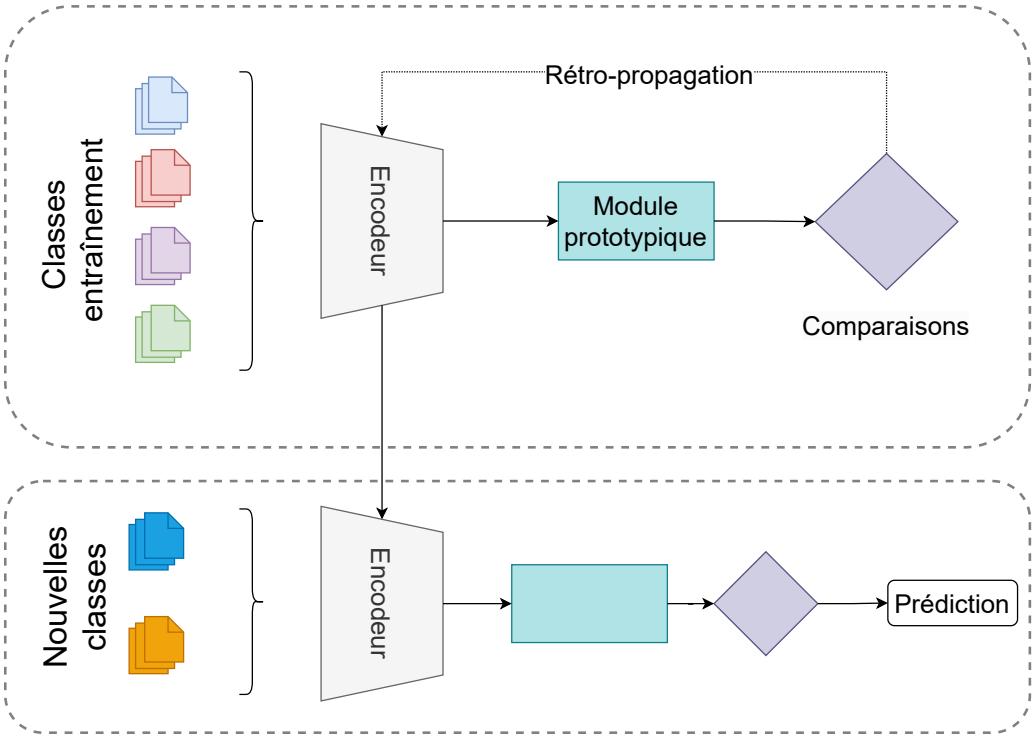


Figure 3.2 – Vue d'ensemble des méthodes prototypiques.

Les méthodes prototypiques standards, par nature non paramétriques [2], accordent une attention particulière à l'amélioration du module d'encodage. En effet, l'objectif principal est de construire des prototypes représentatifs pour chaque classe, capables de saisir au mieux les caractéristiques distinctives de ces classes.

Cette observation a suscité notre intérêt pour l'exploration de techniques visant à enrichir les représentations vectorielles de chaque mot à partir du modèle pré-entraîné BERT(*Bidirectional Encoder Representations from Transformers* (Devlin et al., 2019a)). Nous nous pencherons en particulier sur l'exploitation de combinaisons de couches cachées du modèle pour obtenir des représentations vectorielles plus informatives et discriminantes d'une part, et l'intégration de connaissances à partir d'une tâche de pré-entraînement d'autre part.

La première approche que nous avons explorée repose sur la combinaison des différentes couches cachées du modèle de langue BERT. Nous partons du principe que chaque couche contient des informations à différents niveaux d'abstraction, et donc en combinant ces couches, nous pouvons obtenir des représentations vectorielles plus riches et plus expressives. Par ailleurs, cela peut également être vu comme une méthode d'ensemble dans laquelle chacune des couches représente un encodeur spécifique. De plus, cette méthode ne requiert aucune ressource externe pour sa mise en œuvre puisqu'elle repose sur les informations intrinsèques du modèle.

2. Du point de vue des méta-paramètres.

Dans la seconde stratégie, nous avons cherché à exploiter des connaissances externes pour améliorer les représentations des mots. Cette technique consiste à entraîner le modèle sur une tâche de pré-entraînement supplémentaire, dans le but d'incorporer des informations spécifiques dans les représentations vectorielles des mots. Nous utilisons en particulier l'approche *LexFit* (Vulić et al., 2021), qui vise à construire des représentations proches pour des mots ayant une relation de synonymie et éloigner les mots non-synonymes. En effet, nous estimons que les déclencheurs d'un même type d'événements appartiennent souvent à un même champ lexical à quelques exceptions près (Wang et al., 2021a). Nous avons également exploré une méthode consistant à enrichir les prototypes par des mots clés propres à chaque type d'événements. L'idée sous-tendant cette dernière proposition est qu'à défaut d'avoir des exemples annotés dans un domaine cible, il serait simplement possible de fournir quelques exemples de déclencheurs hors contexte correspondant à ces mots clés.

3.3 . Cadre expérimental

Nous adoptons une approche reposant sur les réseaux prototypiques avec un apprentissage épisodique afin de combiner méta-apprentissage et FSL. Notre objectif est d'améliorer les représentations vectorielles fournies par le module d'encodage pour la détection d'événements à partir de peu d'exemples.

Étant donné une phrase $x = \{w_1, \ldots, w_L\}$, de longueur L, l'objectif de ce module est de construire une représentation h_i de chaque mot $w_i \in x$. Nous utilisons le modèle de langue BERT comme encodeur de base, qui est un modèle de type « transformers » (Vaswani et al., 2017) composé de 12 couches entraînables.

Notre approche repose sur le cadre général composé des quatre modules précédemment présentés. Nous donnons un aperçu du modèle complet à la figure 3.3. En résumé, notre méthode consiste à calculer des plongements de mots à partir de phrase en entrée. Puis à construire les prototypes de chaque classe à partir des déclencheurs contenus dans le support set. Et enfin classifier les exemples du query set en fonction de leurs similarités à ces prototypes.

3.3.1 . Jeu de données

Nous réalisons nos expérimentations sur le corpus FewEvent mis en place par Deng et al. (2020) pour la détection d'événements à partir de peu d'exemples. Ce corpus est composé de 70 852 mentions d'événements, en langue anglaise, réparties en 100 types. Nous utilisons le même découpage que Cong et al. (2021) à des fins de comparaison. Ce découpage comprend 80 types dans l'ensemble d'apprentissage, 10 types dans l'en-

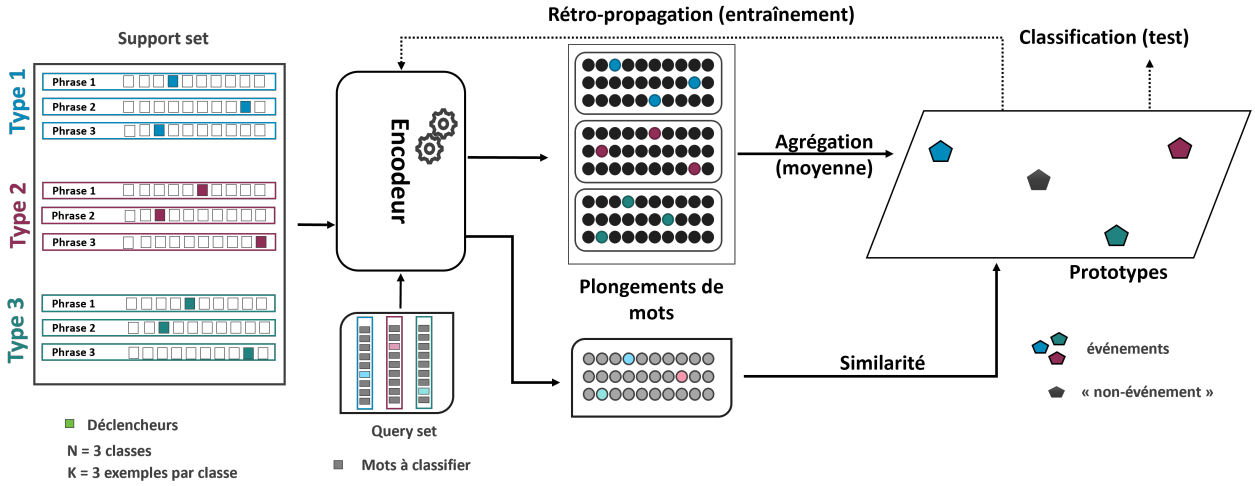


Figure 3.3 – Vue d'ensemble du modèle de détection d'événements à l'aide de réseaux prototypiques. Le modèle commence par prendre des phrases en entrée et génère des plongements de mots à l'aide d'un encodeur. Ensuite, il crée des prototypes d'événements en utilisant les plongements des déclencheurs pour chaque type d'événement, ainsi qu'un prototype pour la classe *NULLE* à partir des mots qui ne sont pas des déclencheurs.

semble de test et les 10 types restants dans l'ensemble de validation. Ce découpage permet d'assurer la distinction entre les classes d'entraînement, de validation et celles d'évaluation.

3.3.2 . Entraînement et évaluation

Nous entraînons notre modèle de façon épisodique et nous mettons à jour les poids de l'encodeur à chaque épisode. Pendant l'entraînement, nous échantillonnons des exemples parmi les 80 classes d'entraînement de façon uniforme.

Nous prenons comme prototype pour chaque classe la moyenne des exemples du support set appartenant à cette classe, comme proposé par Snell et al. (2017). Puisque que nous utilisons le format BIO en entrée, nous construisons un prototype pour les classes `B-` et `I-` ainsi que pour la classe `O` (dite classe *NULLE*), qui désigne les mots qui ne sont déclencheurs d'aucun événement. Au total, nous avons donc $2N + 1$ prototypes pour un épisode composé de N types d'événements.

Pour une séquence donnée, nous calculons la probabilité d'appartenance des tokens à chaque classe en fonction de leur similarité par rapport aux prototypes. Le modèle est ensuite entraîné en utilisant l'entropie croisée (*cross-entropy*) sur cette distribution de probabilités, comme il est d'usage dans les problèmes de classification :

$$\mathcal{L}(w_q, \mathcal{S}) = -\log\left(P(y = k | w_q, \mathcal{S})\right) \tag{3.2}$$

Pour l'évaluation, nous construisons 3 000 épisodes $\mathcal{E}^i \triangleq \{\mathcal{S}^i, \mathcal{Q}^i\}$ avec N types d'événements tirés aléatoirement pour chaque épisode. Nous sélectionnons ensuite k exemples par classe dans le support set et un exemple par classe dans le query set. Les exem-

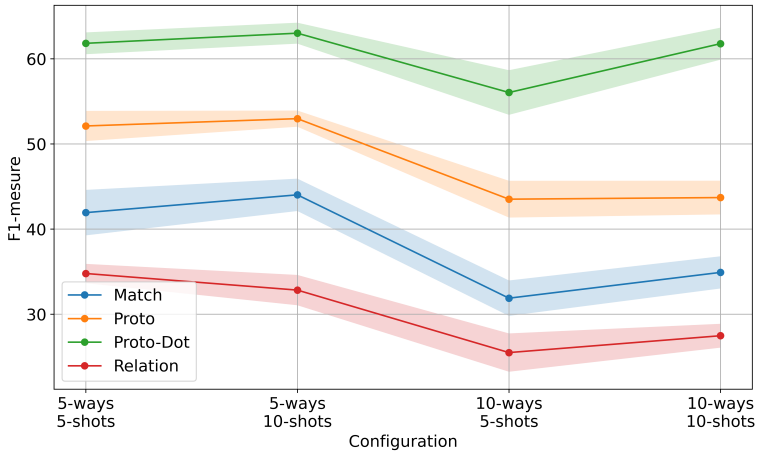


Figure 3.4 – Comparaison des différents modèles de méta-apprentissage. Moyennes et écart-types de la F1-mesure sur cinq expériences.

ples du support set servent à construire les prototypes et les exemples du query set sont classifiés en fonction de leur similarité à ces prototypes. Nous considérons qu'un déclencheur d'événement est correct si son type et sa position dans la phrase sont correctement prédits, comme dans les travaux précédents en détection d'événements (Cong et al., 2021; Cui et al., 2020; Liu et al., 2018a). Nous adoptons la F1-mesure pour l'évaluation des performances comme dans les travaux précédents.

3.3.3 . Comparaison des différentes configurations

Afin de comparer les différentes méthodes prototypiques entre elles, nous avons simplement réimplémenté les travaux de Cong et al. (2021). Les résultats de cette étude préliminaire sont présentés à la figure 3.4.

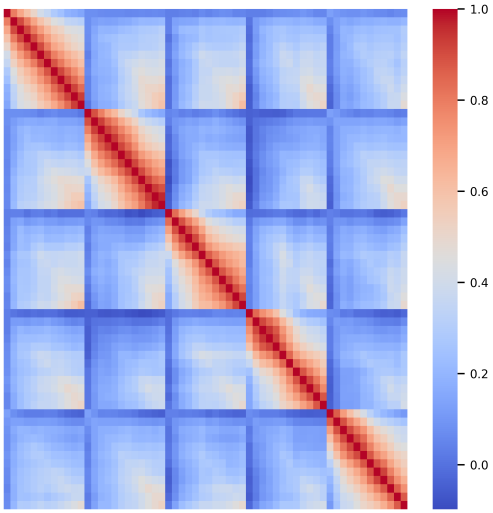
Proto-Dot est une variante des réseaux prototypiques utilisant le produit scalaire comme fonction de similarité à la place de la distance euclidienne. Proto, Match et relation correspondent respectivement aux réseaux prototypiques (*Prototypical Networks*), de correspondance (*Matching Networks*) et de relation (*Relation Networks*).

La figure 3.4 donne les performances de ces modèles dans différentes configurations N-ways, k-shots sur le jeu de données FewEvent. Nous constatons que la version Proto-dot affiche des performances supérieures par rapport aux autres méthodes. Cette observation corrobore les résultats de plusieurs études. Par exemple, dans le domaine de la classification de textes, Dopierre et al. (2021) ont montré que les réseaux prototypiques simples se démarquaient en termes de performances lorsque toutes les méthodes étaient évaluées dans des configurations rigoureusement comparables. Par conséquent, c'est cette configuration que nous adopterons pour évaluer l'impact des enrichissements apportés à l'encodeur dans les sections à suivantes.

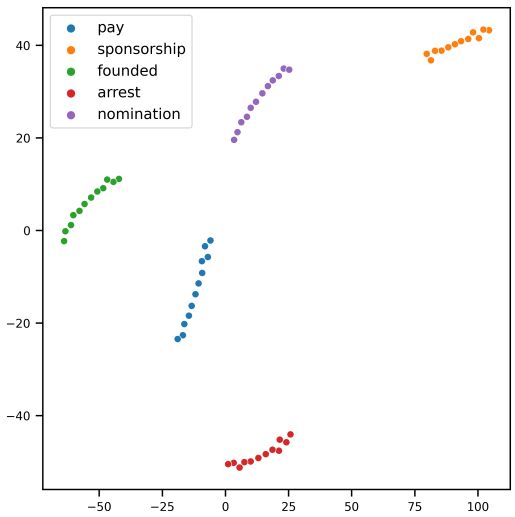
3.4 . Enrichissement par combinaison de couches

La première stratégie que nous employons pour améliorer les représentations des mots repose sur la combinaison des couches cachées du modèle BERT. Ce choix découle de la structure hiérarchique de BERT, qui utilise plusieurs couches « transformers » pour capturer différents niveaux d'abstraction dans le texte. En combinant les sorties de ces différentes couches cachées, nous cherchons à exploiter des informations à différents niveaux de granularité, étant donné que chaque couche contient des informations contextuelles.

Cette stratégie d'amélioration s'appuie sur l'idée que les informations à des niveaux d'abstraction différents peuvent contribuer à la discrimination entre les différentes classes d'événements, mais aussi à une augmentation artificielle du nombre d'exemples sans utiliser de ressource externe. En effet, on peut considérer que les plongements issus des couches cachées sont aussi des exemples potentiels de déclencheurs vu que les représentations issues des couches cachées sont relativement proches des représentations de la couche de sortie pour un mot donné. Pour illustrer ce constat, nous présentons à la figure 3.5 les valeurs de similarité (cosinus) entre les représentations fournies par les 12 couches de BERT pour cinq déclencheurs différents (figure 3.5a) ainsi que leurs plongements visualisés en deux dimensions avec t-SNE (cf. figure 3.5b). On peut voir que, de façon attendue, les plongements des couches cachées pour un même mot sont bien similaires comparés à d'autres mots. On remarque également que plus l'on s'éloigne de la couche de sortie, plus les similarités entre deux déclencheurs différents ont tendance à augmenter. Cela laisse à penser que les premières couches encodent plutôt des informations génériques, moins sensibles à la notion de similarité sémantique entre mots, contrairement aux dernières couches.



(a) Similarité cosinus entre les plongements des différentes couches pour 5 déclencheurs différents.



(b) Plongements de 5 déclencheurs différents sur les 12 couches.

Figure 3.5 – Illustration de la similarité entre les représentations des différentes couches pour cinq déclencheurs différents.

L'idée de combiner les couches avait déjà été suggérée dans l'article de Devlin et al. (2019a), qui montrait que la concaténation des quatre dernières couches de BERT améliorait ses performances sur certaines tâches. De plus, van Aken et al. (2019) ont réalisé une analyse approfondie des couches de BERT sur plusieurs tâches de question-réponse et ont également conclu que ce n'était pas nécessairement la dernière couche qui obtenait les meilleures performances. La figure 3.6 présente les performances obtenues pour chacune des couches de BERT sur cinq tâches différentes, pour un modèle BERT-base à douze couches et un modèle BERT-large à vingt-quatre couches.

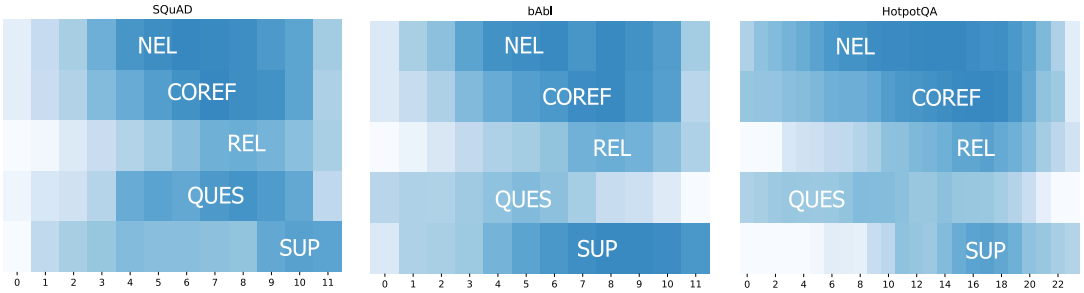


Figure 3.6 – Score d'exactitude (*accuracy*) pour chaque couche de BERT dans cinq tâches sur trois benchmarks. Source : van Aken et al. (2019).

Partant de l'option par défaut appelée **BERT**, nous avons exploré cinq configurations de sélection et de combinaison des couches.

BERT utilise la sortie de la dernière couche de BERT comme plongement du mot $e_i = h_i^{12}$. À la suite de Devlin et al. (2019a), c'est l'option adoptée de façon générale pour les tâches de traitement automatique de langues utilisant BERT, en particulier par Cong et al. (2021) et Lai et al. (2021a) dans notre contexte le plus proche.

Nous construisons ensuite les différentes variantes ci-dessous dans le but d'apporter des améliorations par rapport à cette version standard.

Average prend la moyenne des représentations sur m couches consécutives comme plongement du mot w_i.

$$e_i = \frac{1}{m}\sum_{k=1}^{m} h_i^k \quad \text{ou} \quad e_i = \frac{1}{m}\sum_{k=12-m+1}^{12} h_i^k$$

suivant que les couches sont associées à partir de la première ou de la dernière. Nous n'avons pas évalué les cas où les couches sélectionnées ne sont pas consécutives afin de limiter la combinatoire du problème.

Max-pool fait une agrégation *max-pooling* sur chaque dimension d sur les m couches consécutives considérées. Le p-ième élément du plongement e_i est donné par :

$$(e_i)_p = \max((h_i^1)_p, \ldots, (h_i^m)_p)$$

Concat fait la concaténation des sorties des m couches consécutives considérées.

$$e_i = [h_i^1 \,||\, h_i^2 || \ldots || h_i^m] \quad \text{ou} \quad e_i = [h_i^{12} \,||\, h_i^{11} \,||\, \ldots \,||\, h_i^{12-m+1}]$$

Weighted fait une combinaison linéaire des 12 couches de BERT. L'objectif ici est de faire une moyenne comme avec **Average**, mais avec des pondérations apprises pendant l'entraînement du modèle. Cette méthode permet de donner plus ou moins d'importance à chaque couche en fonction de sa contribution à la résolution de la tâche.

$$e_i = \sum_{k=12}^{12} \alpha^k h_i^k$$

où les α^k sont des poids initialisés aléatoirement et appris.

ATT fait une combinaison linéaire sur chaque dimension par mécanisme d'attention, l'objectif étant d'identifier pour chaque dimension la ou les couches les plus importantes. Le p-ième élément de e_i est donné par

$$(e_i)_p = \sum_{k=1}^{12} \alpha^k (h_i^k)_p$$

où les α^k sont obtenus par apprentissage à partir d'une combinaison linéaire sur les 12 couches et d'une normalisation softmax. $\alpha^k = \text{softmax}(\beta)^k$ où $\beta = W_i H_i + b_i$, $W_i \in \mathbb{R}^d$, et $b_i \in \mathbb{R}^{12}$ étant les poids d'un modèle linéaire appris, et $H_i \in \mathbb{R}^{d \times 12}$ la sortie de toutes les couches du modèle BERT.

3.4.1 . Résultats

Les performances pour chacune de ces combinaisons sont consignées dans le tableau 3.2. Nous avons réalisé toutes les évaluations avec un modèle Proto-dot.

Encodeur	5-ways 5-shots	5-ways 10-shots	10-ways 5-shots	10-ways 10-shots
BERT	61,22 ± 0,90	60,84 ± 1,58	58,14 ± 1,69	59,85 ± 2,01
Average	64,34 ± 1,94	65,37 ± 0,66	61,85 ± 2,05	63,93 ± 1,08
Max-pool	64,10 ± 1,78	65,80 ± 0,91	61,15 ± 1,51	63,37 ± 1,03
Concat	61,99 ± 0,46	61,94 ± 0,97	57,47 ± 1,65	59,02 ± 1,39
Weighted	65,62 ± 1,55	**67,15** ± 0,88*	**62,63** ± 1,18*	**65,22** ± 0,98*
ATT	**65,64** ± 0,90	65,63 ± 0,46	62,22 ± 0,52	64,23 ± 0,99

Table 3.2 – Comparaison des stratégies de combinaison de couches dans différentes configurations N-ways, k-shots avec un modèle Proto-Dot sur 5 essais. La meilleure performance en moyenne est indiquée **en gras**, la deuxième est soulignée. * indique que la différence entre le meilleur modèle et le deuxième est statiquement significative, en utilisant le test de significativité de Dror et al. (2019).

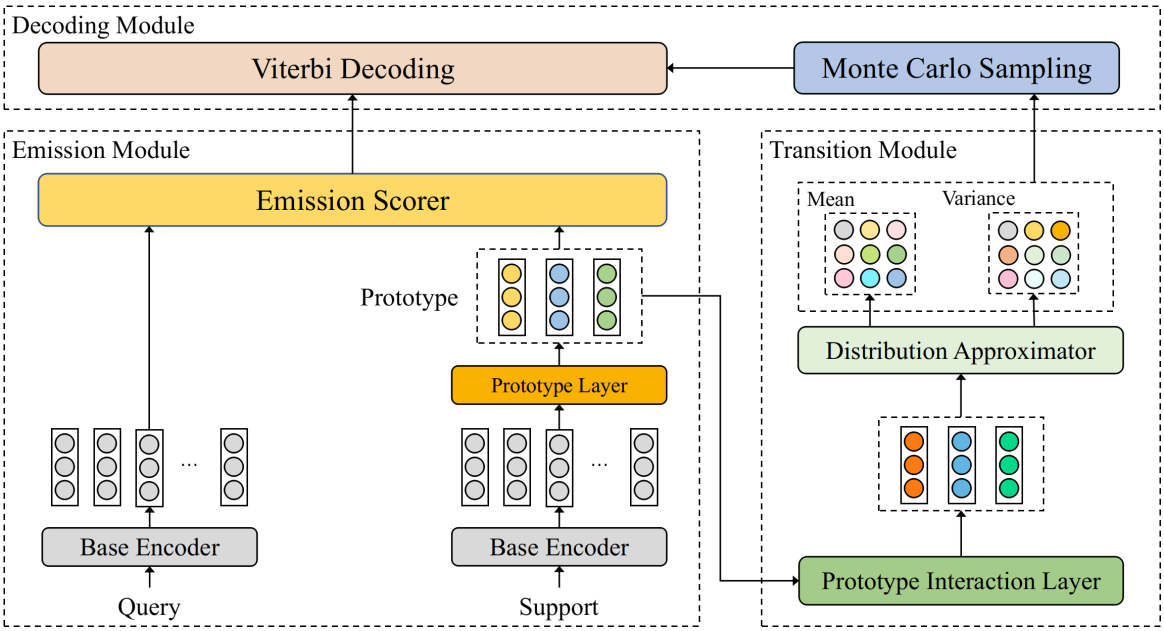


Figure 3.7 – Modèle PA-CRF de Cong et al. (2021). Source : (Cong et al., 2021).

Ces résultats montrent que, quelle que soit la configuration N-ways, k-shots, toutes les modifications de l'encodeur permettent d'améliorer de façon significative les performances par rapport à l'encodeur BERT traditionnel. La combinaison **Weighted** est meilleure que toutes les autres combinaisons sauf pour la configuration 5-ways 5-shots où elle reste très compétitive avec la combinaison **ATT**. Si la version **Weighted** est meilleure que **ATT** la plupart du temps, il faut néanmoins noter que **ATT** obtient des variances bien moins élevées. En effet, **ATT** est moins sensible aux conditions initiales, car les poids des combinaisons sont calculés directement à partir des couches elles-mêmes.

3.4.2 . Comparaison avec l'état de l'art

Afin d'évaluer l'impact des modifications sur l'encodeur, nous utilisons deux modèles présentés dans Cong et al. (2021) : **Proto-dot**, un modèle prototypique utilisant le produit scalaire comme fonction de similarité, qui est notre modèle de référence [3], et **PA-CRF**, une amélioration du modèle précédent utilisant des couches CRF (*Conditional Random Fields*) (Lafferty et al., 2001) pour estimer les probabilités de transition entre les différentes étiquettes BIO comme proposé par Hou et al. (2020). Ce modèle PA-CRF est la contribution principale de Cong et al. (2021) et constituait le meilleur modèle de l'état de l'art en 2021. Les CRF sont un type de modèle probabiliste largement utilisé dans le domaine du traitement du langage naturel pour les tâches d'annotation de séquences permettant de modéliser les dépendances entre les étiquettes au sein d'une séquence. Nous donnons un aperçu du modèle PA-CRF à la figure 3.7.

3. Les résultats de ce modèle sont récapitulés dans le tableau 3.2. Nous présentons ici les résultats obtenus avec l'encodeur BERT standard, ainsi que ceux obtenus avec **Weighted**, notre encodeur le plus performant, à des fins de comparaison.

Nous adoptons la micro F1-mesure pour évaluer les performances et nous rapportons les moyennes et les écart-types sur 5 essais dans le tableau 3.3 avec différentes valeurs de N et k. Pour les encodeurs **Average**, **Concat** et **Max-pool**, nous ne prenons en considération que les 4 dernières couches pour les résultats rapportés dans le tableau.

Nous comparons notre modèle au modèle de Cong et al. (2021), qui était l'état de l'art au moment de ces travaux. Nous avons ré-implémenté la version de Cong et al. (2021) correspondant à la ligne **BERT** et nous avons également rapporté les résultats fournis dans leur article (**BERT [Cong]**).

Modèle	**Encodeur**	**5-ways 5-shots**	**5-ways 10-shots**	**10-ways 5-shots**	**10-ways 10-shots**
Proto-dot	BERT	61,22 ± 0,90	60,84 ± 1,58	58,14 ± 1,69	59,85 ± 2,01
	Weighted	65,62 ± 1,55	**67,15** ± 0,88*	**62,63** ± 1,18*	**65,22** ± 0,98*
PA-CRF	BERT [Cong]	62,25 ± 1,42	64,45 ± 0,49	58,48 ± 0,68	61,54 ± 0,89
	BERT	63,63 ± 2,01	63,66 ± 1,54	62,11 ± 1,58	62,47 ± 1,29
	Average	65,09 ± 0,40	66,70 ± 0,45	62,32 ± 1,51	65,38 ± 1,71
	Max-pool	63,95 ± 1,99	66,94 ± 1,20	61,74 ± 1,95	64,77 ± 1,84
	Concat	64,30 ± 1,99	64,31 ± 1,80	62,01 ± 1,28	61,88 ± 1,05
	Weighted	**66,26** ± 1,16*	**66,97** ± 0,95*	**63,90** ± 1,23*	**67,21** ± 1,27*
	ATT	63,65 ± 1,35	66,40 ± 1,03	62,41 ± 1,73	64,32 ± 1,64

Table 3.3 – Résultats : moyenne et écart-type de la micro F1-mesure sur 5 essais. La meilleure performance en moyenne est indiquée **en gras**, la deuxième est soulignée. * indique que la différence entre le meilleur modèle et le deuxième est statiquement significative, en utilisant le test de significativité de Dror et al. (2019).

Le tableau 3.3 montre d'abord que, quel que soit le modèle utilisé (Proto-dot ou PA-CRF), toutes les modifications de l'encodeur, hormis la configuration **Concat** pour la condition 10-ways de PA-CRF, permettent d'améliorer de façon significative les performances par rapport à l'encodeur BERT classique.

Cette observation permet également de conclure que ces améliorations sont cumulables avec des améliorations dans d'autres modules. En effet, l'apport de PA-CRF se fait au niveau du module de prédiction. Une meilleure exploitation des informations du modèle BERT permet donc de dépasser les améliorations apportées par le modèle plus complexe de Cong et al. (2021), représentant l'état de l'art de l'époque.

Parmi les différentes stratégies testées, celles permettant au système d'apprendre automatiquement les poids pour combiner les différentes couches donnent généralement de meilleurs résultats, la stratégie **Weighted** s'avérant la meilleure dans presque tous les cas.

Une analyse plus détaillée des résultats des différentes approches, en particulier pour les approches **Average**, **Concat** et **MaxPool**, est proposée en section 3.6.4.

3.5 . Enrichissement par injection de connaissances

La seconde méthode d'enrichissement que nous avons envisagée est l'enrichissement par injection de connaissances. L'intégration de connaissances, qu'elles soient lexicales, syntaxiques ou sémantiques, joue un rôle important dans l'amélioration des performances des modèles de langue pré-entraînés. La tâche de pré-entraînement en elle-même peut être vue comme une façon implicite d'injecter des connaissances spécifiques à un domaine dans un tel modèle (Petroni et al., 2019). En particulier, le modèle de fondation BERT a été pré-entraîné à l'origine sur la prédiction de mot masqué (*Masked Language Modeling*, MLM) et la prédiction de la cohérence entre deux phrases données (*Next Sentence Prediction*, NSP). La première tâche permet de former des représentations contextuelles pertinentes pour chaque mot et la seconde, la représentation de la sémantique des phrases. Ces tâches conduisent ainsi à encoder des connaissances sur le domaine dans lequel est réalisé ce pré-entraînement. Une manière intuitive d'adapter ces modèles de fondation à de nouveaux domaines, sans données annotées, consiste à simplement continuer son pré-entraînement de façon auto-supervisée sur le domaine en question. Nous n'avons pas exploré cette piste parce que les jeux de données sur lesquels nous travaillons sont déjà dans le domaine général.

Pour ajouter des connaissances explicites dans le modèle, une des stratégies courantes consiste à exploiter des ressources externes telles que des bases de données lexicales ou des thésaurus sémantiques pour enrichir le vocabulaire du modèle. Cela permet d'introduire une meilleure couverture lexicale, ce qui est crucial pour la détection et la classification précises des événements et de leurs composants. De plus, ces ressources peuvent être utilisées pour renforcer les liens sémantiques entre les entités et les événements, améliorant ainsi la cohérence des prédictions.

3.5.1 . Enrichissement des prototypes par mots clés

L'idée ici est de renforcer les prototypes de chaque classe en y intégrant les représentations de mots potentiellement déclencheurs (comme Bronstein et al. (2015), Lai and Nguyen (2019) et Yu et al. (2022)). En effet, pour se rapprocher d'un cas d'application réel, nous pouvons faire l'hypothèse qu'à défaut d'avoir des exemples annotés, nous pouvons avoir des listes de mots clés associés à chaque nouveau type d'événements.

Sélection des mots clés

Cette idée d'enrichissement par des mots clés a été inspirée par les travaux de Bronstein et al. (2015); Lai and Nguyen (2019); Yu et al. (2022), qui ont exploité le guide d'annotation du jeu de données ACE-2005 pour sélectionner comme mots clés les exemples associés à chaque type d'événement. Étant donné que le jeu de données FewEvent n'a pas fourni de guide d'annotation, nous prenons entre 1 et 5 déclencheurs par classe, de façon aléatoire.

Intégration des mots clés

La méthode proposée consiste simplement à modifier le prototype de chaque classe en y intégrant un prototype des mots clés. Lors de chaque épisode, nous ajoutons la moyenne des représentations des mots clés au prototype de chaque classe. Nous encodons les mots clés de façon isolée (hors contexte) avec le même encodeur BERT que les mentions d'événements. Lorsque le mot clé sélectionné est polylexical, il est représenté par la moyenne des représentations de ses composants. Le nouveau prototype est donné par :

$$\tilde{c}^y = \frac{1}{2}\left(\alpha c^y + \beta c^y_{kw}\right)$$

où c^y et c^y_{kw} sont respectivement le prototype issu du support set et des mots clés. α et β sont des termes de normalisation.

3.5.2 . La méthode LexFit

Les travaux de cette section ont été réalisés par M. Rida Lali dans le cadre de son stage de Master, que j'ai co-encadré avec mes encadrants.

La deuxième méthode que nous explorons pour améliorer les représentations vectorielles de mots est le système LexFit (Vulić et al., 2021), une approche visant à enrichir les représentations lexicales en exploitant les similarités lexicales entre les mots issues d'un dictionnaire de synonymes. Une vue d'ensemble de la méthode est donnée à la figure 3.8.
L'approche LexFit est elle-même inspirée du modèle Sentence-BERT [4] (Reimers and Gurevych, 2019), conçu pour produire des représentations similaires pour des phrases sémantiquement proches. Contrairement aux méthodes traditionnelles qui se concentrent principalement sur l'entraînement de modèles sur de vastes corpus de texte, LexFit exploite des ressources lexicales telles que des dictionnaires de synonymes et d'antonymes pour enrichir les représentations des mots. L'idée fondamentale de LexFit est de combiner les informations sémantiques des ressources lexicales et les représentations contextuelles des mots, généralement produites par des modèles de langage préentraînés comme BERT. En intégrant ces connaissances lexicales dans le processus d'apprentissage, LexFit parvient à améliorer la qualité des représentations vectorielles d'un point de vue lexical, ce qui peut profiter à diverses tâches de traitement automatique des langues, notamment la tâche de détection d'événements. Cet a priori est d'autant plus justifié ici que notre formulation prototypique de la tâche est assez lexicalisée. En

4. https ://github.com/UKPLab/sentence-transformers

principe, nous cherchons simplement à construire un encodeur capable de fournir des représentations de chaque type d'événements et de chaque exemple de sorte que les déclencheurs d'un même type aient des représentations suffisamment proches.

En pratique, pour un mot v_i donné, le modèle prend un autre mot synonyme et un mot non-synonyme pour former respectivement une paire positive et une paire négative. Le modèle est ensuite entraîné de sorte à rapprocher les représentations des mots de la paire positive et éloigner celles de la paire négative à travers une fonction de coût contrastive. Dans notre implémentation, nous n'avons testé que la fonction `MNEG` (*Multiple NEGatives ranking loss*) donnée dans l'équation 3.3. Le modèle prend en entrée des lots (*batch*) de paires afin d'augmenter le signal d'entraînement à chaque itération. Pour des raisons de simplicité, les paires négatives sont construites au sein du lot en sélectionnant simplement un mot d'une autre paire positive au sein du même lot.

$$\mathcal{L}_{\text{MNEG}} = -\sum_{i=1}^{B} s(v_i, w_i) + \sum_{i=1}^{B} \log \left(\sum_{j=1, j\neq i}^{B} \exp s(v_i, w_j) \right) \tag{3.3}$$

où $s(.,.)$ est une fonction de similarité, v_i et w_j des plongements de mots et B la taille du lot.

Le terme $\log \left(\sum_{j=1, j\neq i}^{B} \exp s(v_i, w_j) \right)$, appelé LSE (*Log Sum Exp*) est souvent utilisé pour calculer une approximation lisse de la fonction $\max$ dans le but de rendre les calculs et les optimisations plus stables et différentiables.

En effet, l'on peut borner ce terme comme suit :

$$\exp \max(s(v_i, w_j)) \leq \sum_{j=1, j\neq i}^{B} \exp s(v_i, w_j) \leq \sum_{j=1, j\neq i}^{B} \exp \max(s(v_i, w_j))$$

$$\exp \max(s(v_i, w_j)) \leq \sum_{j=1, j\neq i}^{B} \exp s(v_i, w_j) \leq B \exp \max(s(v_i, w_j))$$

$$\max(s(v_i, w_j)) \leq \log \left(\sum_{j=1, j\neq i}^{B} \exp s(v_i, w_j) \right) \leq \log(B) + \max(s(v_i, w_j))$$

Ce terme permet ainsi d'approximer le maximum des similarités avec les mots non-synonymes tout en restant différentiable. La fonction `MNEG` permet donc de rapprocher les termes synonymes en minimisant le premier terme et d'écarter les mots non-synonymes en maximisant le second terme.

3.5.3 . Expérimentations

Nous avons évalué trois configurations différentes :

- **Standard** est le modèle standard utilisant l'encodeur BERT pré-entraîné.

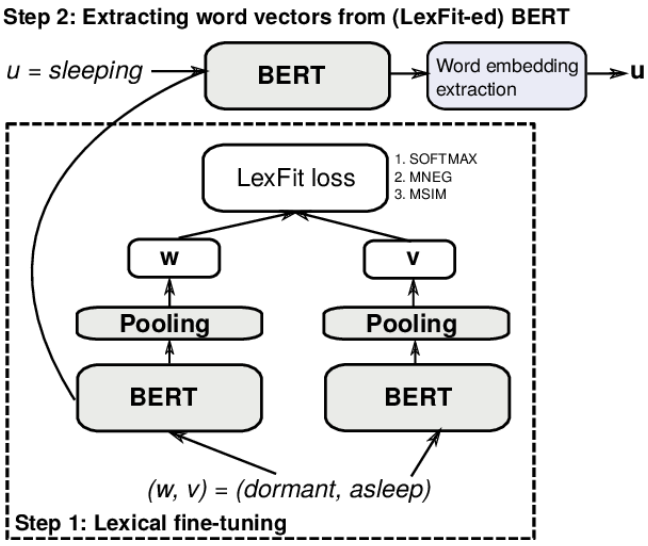


Figure 3.8 – Vue d'ensemble de la méthode LexFit. Source : (Vulić et al., 2021)

- **Mots clés** est la version avec enrichissement des prototypes par les mots clés. Nous avons testé une variante **Mots clés[EVAL]** qui ne considère les mots clés que pendant l'évaluation du modèle. Dans cette variante, le modèle est entraîné sans les mots clés, mais nous ajoutons les mots clés pendant la phase d'évaluation pour enrichir les prototypes des nouvelles classes.
- **LexFit** qui est la version avec le pré-entraînement LexFit.

Résultats

Nous donnons les résultats dans le tableau 3.4. Toutes ces méthodes ont été évaluées avec un modèle Proto-dot dans une configuration 5-ways, 5-shots.

Méthode	F1-mesure
Standard	60,9 ± 2,8
Mots clés	62,7 ± 1,3
Mots clés[EVAL]	62,1 ± 2,2
LexFit	61,6 ± 2,6

Table 3.4 – Comparaison des méthodes par injection de connaissances. Moyennes et écart-types de la F1-mesure sur 5 essais avec un modèle Proto-dot dans une configuration 5-ways, 5-shots.

Nous constatons que toutes ces méthodes d'intégration de connaissances ont un effet positif sur les performances du modèle, bien que cet effet soit moins marqué que celui observé avec les combinaisons de couches. L'intégration de connaissances par le biais de mots-clés semble surpasser l'utilisation de synonymes via LexFit. Cette supériorité peut s'expliquer par la spécificité des mots-clés par rapport aux événements, tandis que les synonymes de LexFit restent d'ordre général. De plus, l'intégration directe des mots-clés dans les prototypes peut jouer un rôle majeur, par opposition à

la méthode LexFit, qui réalise cette intégration de manière préalable via une tâche de pré-entraînement. Néanmoins, il est envisageable de combiner l'intégration de connaissances avec les méthodes de combinaison de couches. Cependant, nos expérimentations dans cette direction ont montré que ces améliorations ne se cumulaient pas de manière significative, les résultats ayant tendance à stagner au niveau de performance atteint par les combinaisons de couches.

3.6 . Discussions

3.6.1 . Impact de la formulation BIO

Nous avons utilisé le format BIO pour la détection et la classification des déclencheurs événementiels, mais ce format pose plusieurs problèmes, en particulier dans un contexte de l'apprentissage à partir de peu d'exemples. Tout d'abord, les méthodes d'apprentissage à partir de peu d'exemples ont des difficultés pour estimer efficacement les transitions entre étiquettes, comme il est d'usage pour les problèmes d'annotation de séquences. C'est d'ailleurs pour cette raison que Hou et al. (2020) proposent une méthode pour apprendre ces transitions. Leur méthode consiste à calculer les scores de transition pour chaque mot en fonction de sa similarité avec la représentation de chaque étiquette au lieu d'apprendre ces transitions. Cette même idée a été reprise dans l'article PA-CRF dans lequel les scores de transition sont calculés à partir des prototypes.

Ce format implique par ailleurs de construire un prototype pour la classe « O », aussi appelée *classe nulle* dans le cadre d'un réseau prototypique. Or, cette classe possède par définition une cohérence très faible car elle est constituée de mots sans lien particulier sur le plan sémantique. La représentativité de son prototype est donc aussi très faible, ce qui tend à perturber les décisions de classification événementielle des mots. Nous étudierons plus en profondeur cette problématique dans le chapitre 4 et proposerons une stratégie permettant de mieux traiter cette classe.

En outre, les classes « `I-` » posent également des problèmes dans la mesure où elles correspondent souvent à des prépositions dans des verbes à particules (*phrasal verbs*) comme on peut le voir dans la figure 3.9. Le modèle a donc tendance à confondre les étiquettes « `I-` » de classes différentes ou à les confondre avec des prépositions appartenant à la classe nulle. De plus, ces étiquettes « `I-` » n'interviennent que dans le cas de déclencheurs en plusieurs mots, qui sont peu représentés dans les corpus (respectivement $0,62\%$, $2,71\%$ et $3,58\%$ dans les ensembles d'entraînement, de validation et d'évaluation du jeu de données FewEvent), ce qui conduit donc à des prototypes peu fiables pour ces classes. Leur impact reste toutefois négligeable sur les performances en raison de leur rareté dans les jeux de données.

Pour remédier à ce problème, nous avons exploré deux approches principales. La première stratégie implique de fusionner les classes « `I-` » avec les classes « `B-` ». En

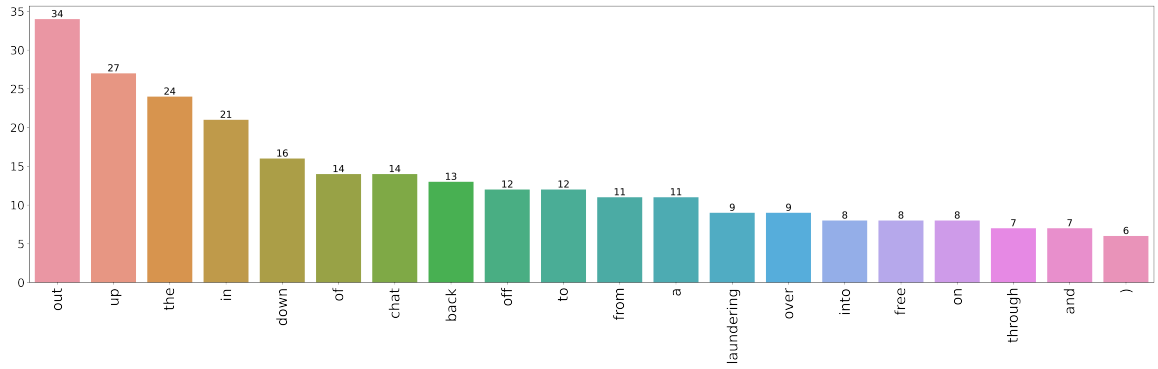


Figure 3.9 – Les 20 mots les plus fréquents avec des étiquettes « I– » dans l'ensemble d'entraînement.

conséquence, tous les mots liés à un même déclencheur seraient regroupés dans une seule classe. Cependant, cette approche entraîne une diminution d'environ 2 points en moyenne des performances du modèle. Dans la seconde approche, nous proposons de créer un prototype « I » commun à toutes les classes « I– ». Cette démarche apporte deux bénéfices significatifs. Tout d'abord, elle aboutit à une réduction du nombre de classes, passant de $(2N+1)$ à $(N+2)$ classes [5]. En outre, elle renforce la représentation du prototype pour cette classe. Ces ajustements ont conduit à une augmentation d'environ 1 point en moyenne de la F1-mesure. Cependant, même si cette méthode permet de réduire les erreurs de prédiction sur les prépositions, elle augmente les erreurs sur les mots « I– » qui n'en sont pas.

3.6.2 . Impact de l'apprentissage épisodique *N-ways, k-shots*

De façon attendue, on remarque que la difficulté de la tâche augmente lorsque le nombre de classes (N) augmente et, de façon complémentaire, que les résultats sont meilleurs avec un plus grand nombre d'exemples annotés (k). En revanche, le nombre d'épisodes d'évaluation n'a pas d'influence significative sur les résultats. Nous avons ainsi vérifié, en faisant varier le nombre d'épisodes de test (avec l'encodeur **Average**) entre 500 et 5 000, que les écarts de scores observés sont de moins de 0,5 point, bien en dessous des écart-types observés dans le tableau des résultats (voir tableau 3.5). Cela démontre que les performances des modèles ne sont pas affectées par le nombre d'épisodes d'évaluation, évitant ainsi un potentiel biais dans l'évaluation qui serait dû à cette évaluation épisodique. De plus, on peut en tirer parti en diminuant le nombre d'épisodes d'évaluation sans perte de généralité.

#épisodes	500	1 000	1 500	2 000	2 500	3 000	3 500	4 000	4 500	5 000
F1-mesure	64,51	64,44	64,83	64,78	64,91	64,74	64,72	64,81	64,69	64,78

Table 3.5 – F1-mesure en fonction du nombre d'épisodes d'évaluation.

5. Représentant toutes les classes « B– », la classe « O », et la classe générique « I ».

3.6.3 . Analyse des erreurs de prédiction

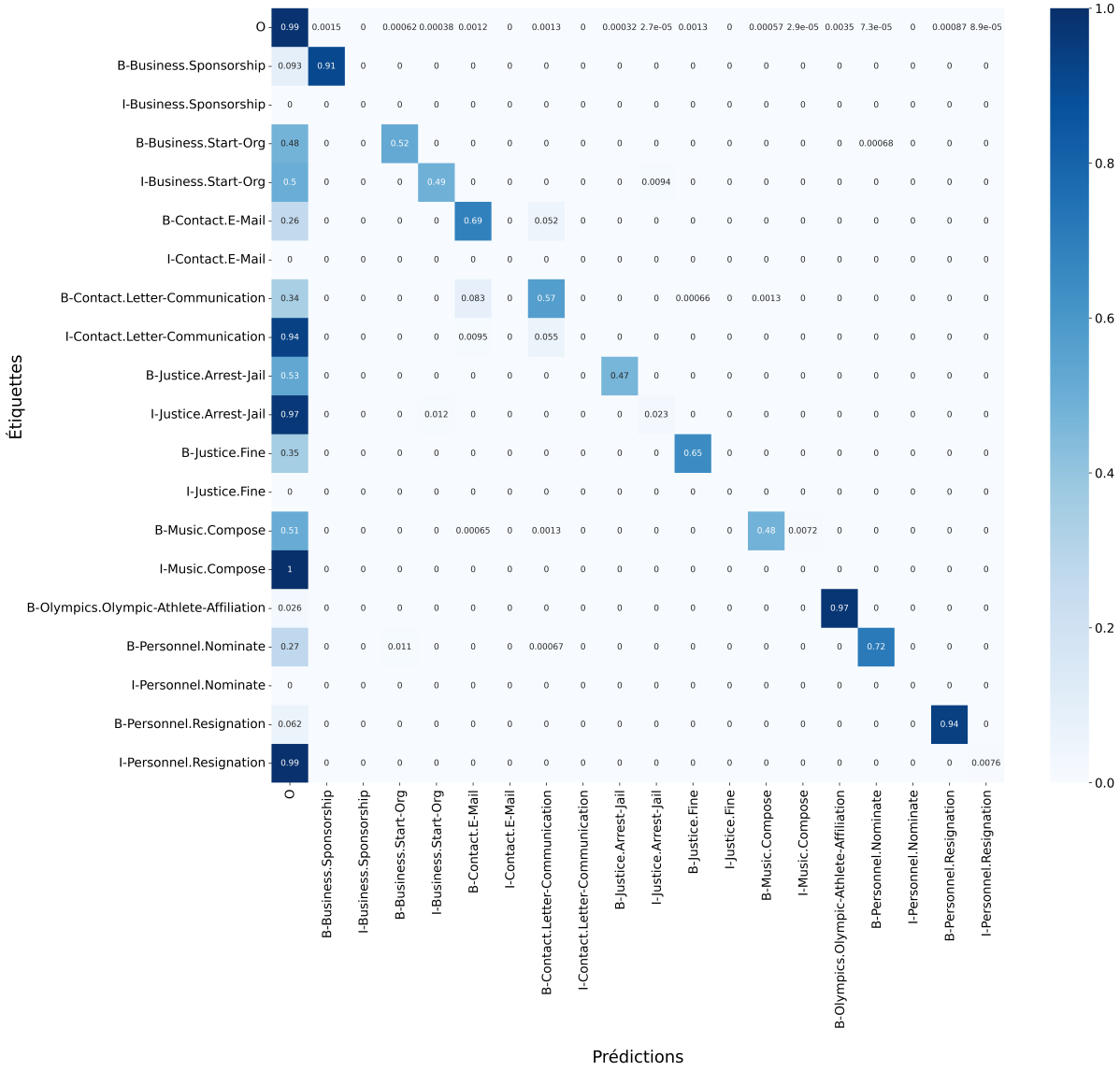


Figure 3.10 – Matrice de confusion pour la détection d'événements sur l'ensemble d'évaluation avec un modèle Proto-dot et l'encodeur Bert-Weighted.

La matrice de confusion de la figure 3.10 donne un aperçu global des erreurs de prédiction faites par le modèle. De manière générale, les classes `B-` sont assez bien prédites. Cela signifie que le modèle arrive à représenter correctement le champ lexical de chaque type d'événements. On peut tout de même remarquer des confusions entre des classes sémantiquement proches[6]. Cela confirme bien l'observation précédente puisque ces classes partagent le même type parent (ici `Contact`) ainsi que des déclencheurs en commun (tels que les verbes *to write* ou *to send*), et donc le même champ lexical.

La plus grande partie des erreurs vient de confusions entre la classe « `O` » (dite classe *NULLE*) et les classes d'événements (première ligne et première colonne de la matrice

6. Par exemple, `Contact.Email` et `Contact.Letter-Communication`.

de confusion). Les faux positifs [7] viennent de ce que le modèle classifie comme non-déclencheurs certains mots qui ont la même orthographe que des mots déclencheurs. Inversement, les faux négatifs [8] proviennent du fait que certains déclencheurs sont trop rares ou peu spécifiques de leur classe et sont donc considérés comme des mots de la classe nulle. Ce problème peut être résolu selon deux voies : d'abord en construisant de meilleurs prototypes pour chaque type d'événements, capables de capter tous les déclencheurs, même les plus rares, pour diminuer le nombre de faux négatifs ; ensuite en trouvant une façon de filtrer les mots « O » classifiés comme déclencheurs. Ces biais ont été définis par Wang et al. (2021a) dans le cadre de la classification d'événements comme étant le problème de chevauchement de déclencheurs (*trigger overlap*) et le problème de séparabilité au sein des déclencheurs (*trigger separability*).

Ces mêmes problèmes ont également été abordés par Chen et al. (2021b), qui proposent de trouver un équilibre entre la représentation du contexte et les caractéristiques intrinsèques des mots dans la construction des prototypes. Cette approche repose sur le fait que l'encodeur BERT capture la signification de la phrase complète par le biais du jeton `[CLS]`, ce qui signifie que le contexte seul peut déjà contenir des informations sur le type d'événement mentionné dans la phrase. Ainsi, lorsque le type d'événement est connu à l'avance, il devient possible de résoudre l'ambiguïté quant à la pertinence d'un mot comme déclencheur ou non.

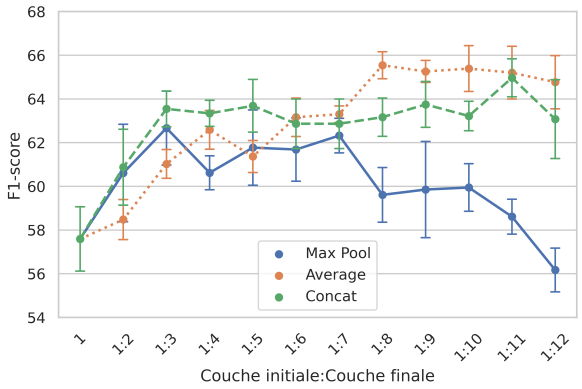
Cependant, leur méthode présente une limitation dans le sens où les prédictions sont effectuées type par type. En pratique, cela équivaut à réaliser une classification binaire avec le contexte seul : déterminer si la mention en question appartient ou non au type traité. Ensuite, pour chaque mot, décider si ce mot agit comme déclencheur ou non. Ces deux étapes de classification binaire sont réalisées conjointement, la seconde étant fortement conditionnée par la première. Leur travail se distingue de l'approche de Bronstein et al. (2015), qui effectue également une évaluation type par type avec une classification binaire au niveau des mots, mais sans prendre en considération le contexte dans lequel le mot apparaît.

Nous proposons une approche permettant de traiter ce problème dans le chapitre 4 en prenant en compte tous les types à la fois.

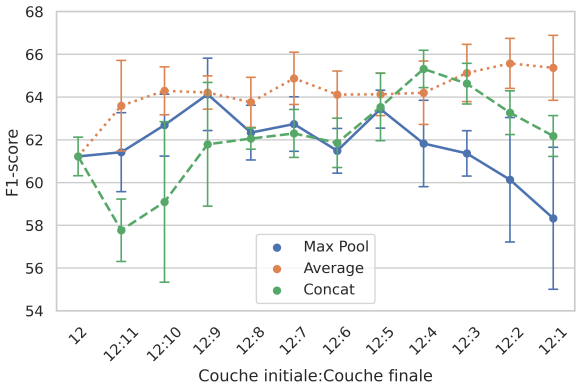
3.6.4 . Combinaison des couches : analyse détaillée

Pour compléter nos analyses, nous avons réalisé des tests afin de déterminer l'influence du nombre de couches sélectionnées pour les encodeurs **Average**, **Concat** et

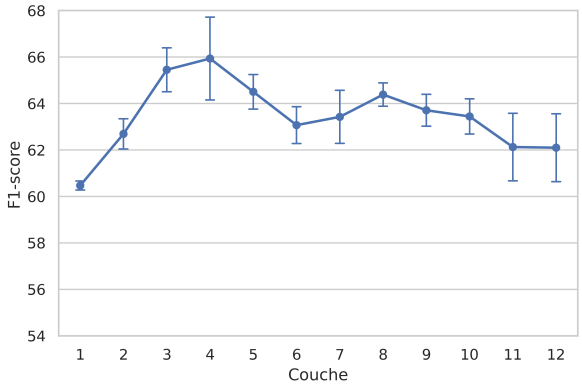
7. Ici, nous désignons par faux positifs tous les mots de la classe « O » assignés à des classes d'événements.
8. Ici, les faux négatifs sont les déclencheurs assignés à la classe « O ».



(a) Premières couches



(b) Dernières couches



(c) Performance couche par couche

Figure 3.11 – Les figures 3.11a et 3.11b présentent l'influence du nombre de couches sélectionnées pour les encodeurs **Average**, **Concat** et **Max-pool** sur les performances du modèle. La figure 3.11c présente les performances du modèle pour une couche isolée.

Max-pool[9]. Nous rapportons à la figure 3.11 les résultats obtenus par ces trois modèles en prenant en compte n couches successives en partant de la première couche (figure 3.11a) ou de la dernière (figure 3.11b).

Nous constatons que l'encodeur **Average** est plus stable que les deux autres et que ses performances ont tendance à augmenter régulièrement avec le nombre de couches. Les meilleurs résultats obtenus avec cette stratégie de combinaison sont par ailleurs compétitifs par rapport à la stratégie **Weighted**, ce qui montre que la prise en compte de toutes les couches reste intéressante, même à l'aide d'une combinaison simple comme la moyenne. Les autres stratégies ne permettent pas, quant à elles, d'exploiter toutes les informations. Cela est particulièrement notable pour la stratégie **Max-pool**, qui tend probablement à lisser de plus en plus les éléments discriminants quand le nombre de couches augmente. En effet, cette approche implique de sélectionner la valeur maximale pour chaque dimension à travers toutes les couches. Ainsi, à mesure que le nombre

9. Étant donné que les stratégies **Weighted** et **ATT** utilisent toutes les couches.

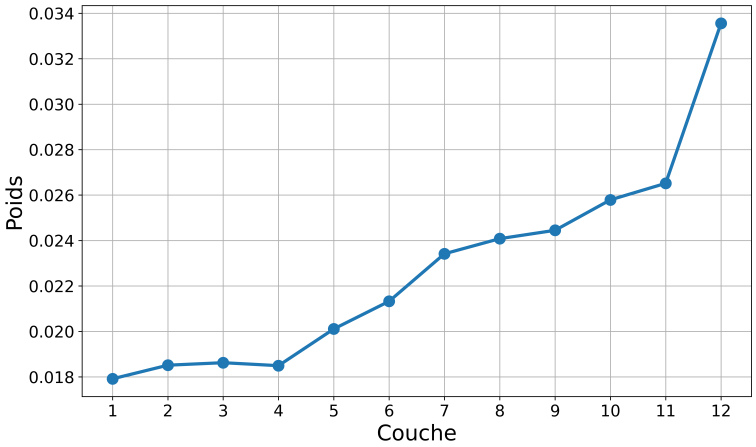


Figure 3.12 – Poids associés à chaque couche dans la combinaison **Weighted**.

de couches s'accroît, ces valeurs maximales auront tendance à se rapprocher pour des mots différents. Cela peut conduire à une perte de spécificité et à une réduction de la capacité à distinguer des caractéristiques subtiles entre les mots.

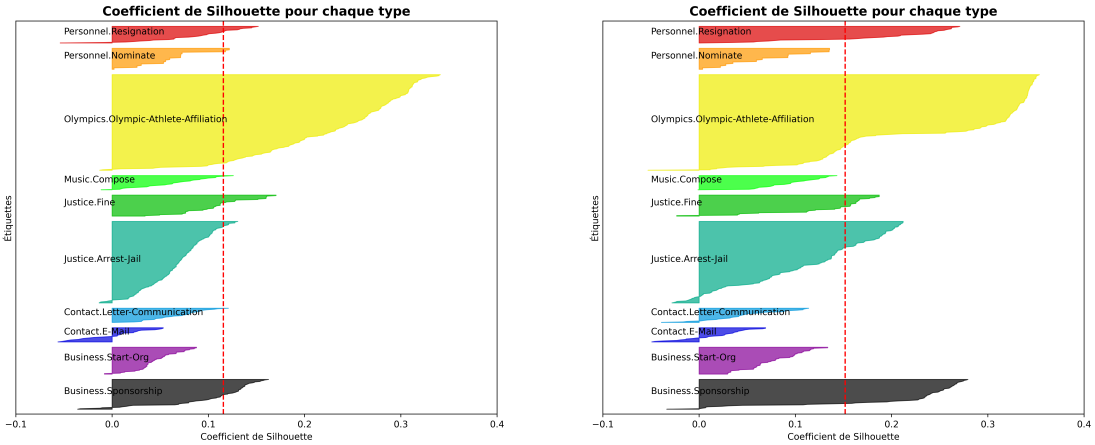
Concat crée pour sa part des représentations intermédiaires de taille importante qui sont sans doute trop peu sélectives du point de vue de leur exploitation par le modèle.

Par ailleurs, les figures 3.11a et 3.11b montrent que les dernières couches de BERT semblent plus intéressantes que les premières dans notre contexte. Ce constat peut traduire la présence intrinsèque d'informations plus utiles pour la tâche visée dans ces couches ou simplement s'expliquer par une influence plus importante de l'apprentissage liée à la tâche à leur niveau du fait de leur plus grande proximité vis-à-vis de la sortie du modèle.

Enfin, la figure 3.11c rapporte le cas limite de la prise en compte d'une seule couche, avec, de la première à la dernière couche, une forte augmentation des résultats jusqu'à la couche 4, qui atteint une performance comparable à **Average** mais avec une variance plus importante, puis une baisse progressive.

Concernant la stratégie **Weighted**, nous avons observé que la combinaison linéaire apprise dépend fortement des poids initiaux et des classes initialement vues par l'encodeur. Afin de regarder qualitativement l'impact de chaque couche, nous avons tracé les poids associés à chaque couche à la figure 3.12. Nous initialisons tous les poids à une même valeur au début de l'entraînement égale à $\frac{1}{12}$. On peut constater que même si le modèle prend en compte toutes les couches, il a tendance à accorder plus d'importance aux dernières couches. Nous pouvons par exemple noter la distinction de la dernière couche par rapport aux autres. Cette tendance n'est plus observée lorsque nous initialisons les poids avec d'autres valeurs.

3.6.5 . Analyses qualitatives



(a) BERT standard

(b) BERT **Weighted**

Figure 3.13 – Coefficients de silhouette pour les encodeurs BERT et BERT-weighted. Les pointillés rouges représentent la valeur moyenne.

Pour évaluer la qualité des représentations vectorielles pour chaque classe, nous avons utilisé l'indice de silhouette, une métrique couramment utilisée en *clustering* pour mesurer à quel point les exemples d'un *cluster* sont similaires entre eux par rapport aux autres *clusters*. L'indice de silhouette varie de -1 à 1. Une valeur élevée indique une séparation claire entre les classes tandis qu'un coefficient faible suggère un chevauchement significatif entre les classes. Les valeurs négatives correspondent aux cas où les exemples sont assignés à la mauvaise classe.

Nous avons tracé les coefficients de silhouette pour les représentations fournies par BERT et **Weighted** afin de comparer leur qualité. En examinant ces coefficients de silhouette, il est notable que la séparation entre les différents types d'événements n'est pas particulièrement nette. Les valeurs moyennes enregistrées se situent à des niveaux relativement bas (0.11 pour BERT et 0.16 pour Weighted), ce qui souligne la complexité inhérente à la tâche de détection d'événements. Cependant, on peut observer que, de façon attendue, les enrichissements apportés par l'approche **Weighted** semblent avoir amélioré la qualité des représentations.

3.7 . Conclusions

Dans ce chapitre, nous avons abordé la tâche de détection d'événements à partir de peu d'exemples. Cette tâche consiste à identifier les déclencheurs d'événements au sein des phrases et leur associer un type. Inspiré par la littérature sur cette même tâche, nous l'avons formulée comme une tâche d'annotation de séquences et nous la traitons par le biais du méta-apprentissage et par une approche prototypique qui a montré des performances prometteuses dans plusieurs études précédentes (Deng et al., 2020; Lai et al., 2020a, 2021a; Cong et al., 2021). En particulier, nous avons cherché à améliorer

les représentations vectorielles des mots fournies par le modèle de langue BERT afin d'améliorer la représentation de chaque prototype et ainsi améliorer les prédictions qui se font sur la base des similarités avec ces prototypes. Dans ce contexte, nous avons exploré différentes stratégies permettant d'améliorer ces représentations. Nous avons évalué d'une part une méthode par **combinaison des couches cachées** du modèle de langue BERT, et d'autre part, une méthode par **injection de connaissances**.

La stratégie d'enrichissement par combinaison de couches repose sur l'idée que les couches cachées du modèle BERT contiennent des informations qui ne sont pas toujours exploitées directement dans les applications. Cette idée avait été déjà évoquée dans l'article initial de BERT, qui suggérait l'utilisation de concaténations de couches dans certaines circonstances. Lorsqu'il y a suffisamment de données pour apprendre, on peut supposer que les informations des couches cachées sont implicitement transférées vers la couche de sortie grâce au mécanisme d'apprentissage. Cette technique devient particulièrement pertinente dans un contexte à faible quantité de données, où l'on peut interpréter les plongements issus de ces couches cachées comme des exemples supplémentaires utiles à la résolution de la tâche.

La méthode par injection de connaissances se décline en deux stratégies. D'une part, l'injection de connaissances spécifiques aux événements à travers des listes de mots clés permettant d'enrichir les prototypes. D'autre part, l'injection par une méthode de pré-entraînement, LexFit (Vulić et al., 2021), fondée sur la notion de synonymie entre mots et permettant de partir d'un modèle pré-entraîné plus apte à reconnaître des similarités lexicales.

Nous avons montré que toutes ces stratégies d'enrichissement permettent d'apporter des améliorations par rapport à l'utilisation standard du modèle BERT pré-entraîné d'origine. En outre, ces améliorations ont permis de surpasser les performances de l'état de l'art pour cette tâche, donnant lieu à deux publications scientifiques (Tuo et al., 2022b,a). Compte tenu du faible volume de données annotées disponibles, l'exploitation optimale des informations fournies par le modèle de langue devient particulièrement intéressante.

Toutefois, malgré les gains de performance significatifs obtenus grâce à notre approche par rapport à l'état de l'art, notre méthode présente encore certaines limitations. Les performances globales de notre modèle demeurent en deçà de celles obtenues dans un cadre supervisé. Cette observation peut être attribuée à la nature intrinsèquement complexe de la tâche ainsi qu'à la formulation prototypique que nous avons adoptée. En effet, notre approche prototypique, fondée sur la similarité entre déclencheurs, peut rendre difficile l'identification de déclencheurs qui diffèrent significativement de la majorité des déclencheurs d'une classe donnée. De plus, la création d'un prototype pour la classe des mots non-déclencheurs (classe *NULLE*) peut poser problème, car ces mots

n’ont pas nécessairement de similarité sémantique intrinsèque. Nous allons consacrer le chapitre suivant au traitement de cette classe nulle en permettant d’améliorer significativement les performances.

Ces résultats sont prometteurs et ouvrent la voie à des améliorations potentielles dans d’autres modules du modèle. En effet, notre étude montre aussi la complémentarité de ces améliorations avec d’autres approches de nature différente, bien que les améliorations induites par les méthodes d’injection de connaissances tendent à s’atténuer lorsqu’on les associe avec les combinaisons de couches.

Chapitre 4

Amélioration de la détection d'événements par une meilleure gestion de la classe *NULLE*

Sommaire

Comme nous l'avons évoqué dans le chapitre précédent, les efforts de recherche ont abordé la détection d'événements à partir de peu d'exemples comme une tâche d'annotation de séquences, qui se transforme en un problème de classification de mots traité à l'aide de réseaux prototypiques (Cong et al., 2021; Tuo et al., 2022b; Zhang et al., 2022c). Toutefois, nous avons observé que cette approche, bien qu'étant prometteuse et à l'état de l'art actuel, est limitée, notamment par le fait qu'elle génère beaucoup de

faux positifs dans ses prédictions. En d'autres termes, les mots non-déclencheurs sont souvent classifiés comme déclencheurs en raison de leur similarité avec les prototypes des classes d'événements.

Dans ce chapitre, nous nous focalisons spécifiquement sur la classe des mots non-déclencheurs, aussi appelée classe *NULLE*, avec pour objectif de réduire de manière significative les erreurs associées à cette classe. Nous explorerons en particulier deux approches principales : la première vise à redéfinir et ajuster le prototype de la classe *NULLE* tandis que la seconde cherche à filtrer cette classe à l'aide d'un seuil, sans construire un prototype spécifique pour celle-ci.

4.1 . Introduction

4.1.1 . Problématique

Les travaux sur la détection d'événements à partir de peu d'exemples abordent la tâche par un apprentissage épisodique N-ways, k-shots (Vinyals et al., 2016) avec des réseaux prototypiques. La tâche de détection des déclencheurs est formulée comme un problème d'annotation de séquence en s'appuyant sur le format BIO (*Beginning-Inside-Outside*), comme dans Cong et al. (2021) et Tuo et al. (2022b)[1]. L'identification du type d'événement et de la position du déclencheur est effectuée en attribuant une étiquette à chaque mot. Cette approche se traduit par une classification multi-classes au niveau des mots, avec autant de classes que de types d'événements, plus une classe dite classe *NULLE* (correspondant aux étiquettes « O ») pour les mots non-déclencheurs. Nous fournissons un exemple d'annotation BIO dans le tableau 3.1 du chapitre précédent.

Ces travaux construisent un représentant pour chaque classe, appelé prototype, à partir des exemples du *support set* en prenant la moyenne des représentations des k déclencheurs fournis. Ensuite, ils classent chaque mot du *query set* en fonction de sa similarité avec ces prototypes (Yang and Katiyar, 2020; Cong et al., 2021; Tuo et al., 2022b).

Bien que cette formulation ait montré des performances prometteuses, comme illustré dans les travaux de Cong et al. (2021) ainsi que dans les expériences présentées dans le chapitre 3, une de ses limitations réside dans sa propension à générer des confusions entre les déclencheurs d'événements et les mots qui ne sont pas des déclencheurs. En conséquence, les performances de ce modèle se traduisent souvent par un rappel élevé, ce qui indique sa capacité à repérer de nombreux déclencheurs, mais s'accompagne d'une précision relativement faible. Autrement dit, il identifie un grand nombre de déclencheurs mais une proportion importante d'entre eux est incorrecte. Nous fournissons une analyse plus fine des résultats par catégorie d'événements dans le tableau 4.1.

1. Les déclencheurs événementiels peuvent être en plusieurs mots, en très faible nombre dans les corpus, mais n'admettent pas de chevauchement, ce qui rend le format BIO suffisant.

Cette modélisation implique d'avoir un prototype pour la classe *NULLE,* qui est construit en rassemblant des mots qui ne sont pas sémantiquement homogènes. L'hétérogénéité intrinsèque du prototype « non-événement » (c'est-à-dire la classe *NULLE*) rend difficile la discrimination entre les mots déclencheurs et certains mots non-déclencheurs qui en sont sémantiquement proches.

Type	Précision	Rappel	F1-mesure
Business.Sponsorship	0,77	0,91	0,83
Business.Start-Org	0,49	0,46	0,48
Contact.E-Mail	0,54	0,78	0,64
Contact.Letter-Communication	0,72	0,64	0,68
Justice.Arrest-Jail	0,78	0,51	0,62
Justice.Fine	0,47	0,72	0,57
Music.Compose	0,66	0,57	0,61
Olympics.Olympic-Athlete-Affiliation	0,37	1,00	0,54
Personnel.Nominate	0,96	0,78	0,86
Personnel.Resignation	0,72	0,92	0,81
moyenne (micro)	0,60	0,73	0,66

Table 4.1 – Précision, Rappel et F1-mesure par classe. Ces résultats sont obtenus avec un modèle Proto-dot, un encodeur **Bert-Weighted** et 5 exemples par classe sur un essai. Nous considérerons ces résultats comme valeurs de référence pour les comparaisons futures.

On peut constater dans le tableau 4.1 que les scores de rappel sont, en général et en moyenne, bien plus élevés par rapport aux scores de précision. Cela signifie que la majorité des déclencheurs est bien identifiée mais qu'il y a également trop de faux positifs. Ce constat est également illustré par la figure 4.1, dans laquelle nous avons tracé les taux pour différents types d'erreur. Nous constatons que la grande majorité des erreurs vient des erreurs de prédiction concernant les étiquettes O. Il est donc nécessaire de mieux modéliser cette classe « O ».

4.1.2 . Traitement spécifique de la classe *NULLE*

Ce problème de classification d'exemples de la classe *NULLE* est souvent inhérent à toutes les tâches d'extraction d'information. Par exemple, pour la tâche de reconnaissance des entités nommées, le fait d'identifier des entités d'intérêt revient également à identifier les autres mots comme n'étant pas des entités. De même, pour la classification de relations, si l'on cherche à classifier des relations particulières entre certaines entités, la plupart d'entre elles n'ont pas forcément de relation entre elles. Dans le cadre de l'extraction d'information supervisée, les travaux considèrent en général cette classe comme une classe supplémentaire et la traite comme les autres classes. Pour les tâches d'annotation de séquences, telles que la reconnaissance d'entités nommées ou la détection d'événements, on tire souvent parti d'une couche CRF pour modéliser les transitions

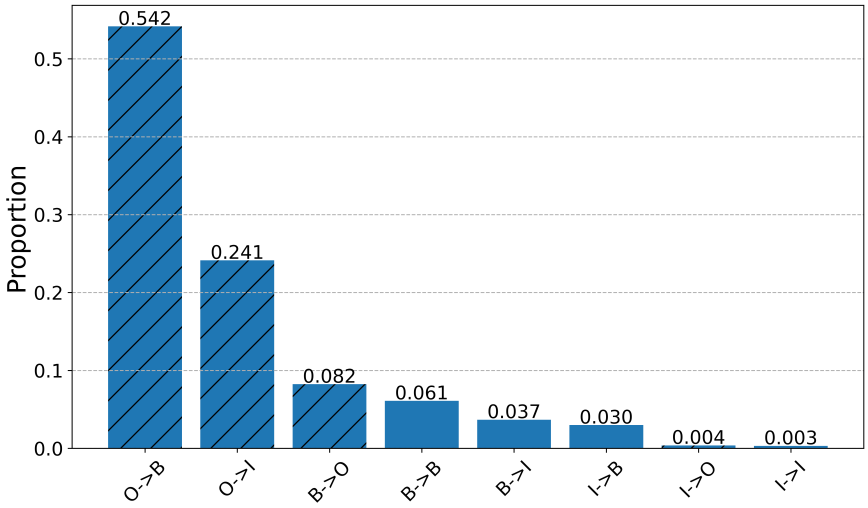


Figure 4.1 – Proportion des types d'erreurs pour un modèle Proto-dot. Les barres hachurées concernent les erreurs liées à la classe *NULLE*.

entre les classes. Cela permet ainsi de s'affranchir du calcul d'une représentation explicite pour cette classe. Cependant, cela ne peut pas être appliqué dans le cadre d'un apprentissage avec peu de données car nous n'aurions pas assez de données pour apprendre ces transitions. En particulier, dans le cadre du méta apprentissage épisodique, les transitions apprises lors de l'entraînement sont peu transférables aux classes d'évaluation vu que ces classes n'apparaissent pas lors de l'entraînement par définition.

Plusieurs travaux ont été proposés pour résoudre ce problème, parmi lesquels (Fritzler et al., 2018) pour la NER et (Gao et al., 2019b) pour l'extraction de relations à partir de peu d'exemples. Fritzler et al. (2018) proposent une méthode ne construisant pas de prototype pour la classe *NULLE* mais définissant la distance à cette classe par une valeur fixe apprise pendant l'entraînement. Une fois cette valeur ajustée, ils font leurs prédictions à l'aide d'une couche CRF qui prend en entrée les distances aux prototypes concaténées avec cette valeur apprise. Gao et al. (2019b) proposent pour leur part le modèle **BERT-PAIR**, qui prédit un score de similarité et un score de dissimilarité pour chaque paire d'instances. Une instance peut donc être catégorisée dans la classe *NULLE* si en moyenne son score de dissimilarité est plus élevé que son score de similarité aux instances du support set.

Dans ce chapitre, nous proposons d'explorer deux types de solutions : une première approche consistant à redéfinir le prototype de la classe nulle et une seconde, consistant à ne pas construire de prototype de cette classe et calculer un seuil de décision pour filtrer les faux positifs.

Ces travaux ont abouti à une nouvelle façon de traiter efficacement la classe *NULLE* dans la tâche de détection d'événements à partir de peu d'exemples. Nous proposons

notamment un nouveau modèle pour la détection d'événements à partir de peu d'exemples en utilisant des réseaux prototypiques et une fonction de coût contrastive par charnière (*Hinge loss*). Nous calculons un seuil de décision dynamique en utilisant la fonction de répartition estimée de façon empirique sur les données (*Empirical Cumulative Density Function*, ECDF). Ces contributions se traduisent par des gains significatifs, évalués sur trois jeux de données, que nous avons présentés dans (Tuo et al., 2023b).

4.2 . Traitement de la classe *NULLE* par redéfinition de son prototype

Nous avons d'abord envisagé des approches visant à redéfinir le prototype de la classe *NULLE* afin de mieux saisir les caractéristiques des mots de cette classe. Pour ce faire, nous nous sommes inspirés de la littérature et avons exploré deux principales pistes.

La première piste s'inspire de l'article de Fritzler et al. (2018), qui a utilisé une méthode similaire pour la tâche de reconnaissance d'entités nommées. L'idée sous-jacente est de considérer que cette classe possède des caractéristiques stables et distinctives, indépendamment des variations sémantiques entre les mots. En adoptant cette méthode, nous cherchons à mieux représenter la classe *NULLE* en utilisant un prototype invariant entre les épisodes.

La deuxième méthode que nous avons explorée consiste à construire plusieurs prototypes pour la classe *NULLE* en utilisant une méthode de clustering. Cette approche s'inspire de l'idée que la classe *NULLE* pourrait être plus diversifiée et que différents sous-groupes de mots pourraient s'y trouver. En appliquant un algorithme de clustering sur les exemples de la classe *NULLE* du support set, nous cherchons à construire plusieurs prototypes pour cette classe, chacun représentant un sous-groupe de mots similaires.

4.2.1 . Prototype *NULL* constant

La première approche que nous avons examinée, **StableProto**, consiste à déterminer une valeur invariante pour le prototype de la classe *NULLE* au lieu de le calculer en utilisant les exemples du support set à chaque épisode. Cette idée a été inspirée par les recherches de Fritzler et al. (2018) mais avec une différence notable : au lieu de régler la distance par rapport au prototype comme Fritzler et al. (2018), nous apprenons à fixer le prototype lui-même. Dans notre démarche, nous avons exploré deux variantes de cette approche.

La première variante, **Proto-Zero**, implique l'utilisation d'un prototype constant, égal au vecteur nul. Contrairement à l'approche classique, où le prototype est adapté aux

caractéristiques du support set, il reste ici fixe avec la valeur zéro tout au long de l'entraînement. En fixant le prototype à zéro, notre processus d'apprentissage s'attache à diriger les mots qui ne sont pas des déclencheurs vers cette valeur constante.

La seconde variante, **Proto-Learn**, un peu moins radicale, cherche à apprendre ce prototype invariant en l'initialisant de manière aléatoire et en le mettant à jour pendant le processus d'entraînement. Contrairement à la première variante, le prototype n'est ici pas figé à zéro mais évolue en réponse aux données d'entraînement. Après l'entraînement, ce prototype reste fixe pendant tous les épisodes d'inférence. Cette approche offre davantage de flexibilité tout en permettant au modèle d'apprendre une représentation adaptée à la classe *NULLE* au fur et à mesure de son exposition aux exemples.

La principale innovation de cette formulation réside dans son détachement du besoin de construire le prototype à partir du support set. Cette approche contourne ainsi le défi lié à la diversité des mots de la classe *NULLE* en les attirant vers un vecteur constant ou en permettant une adaptation progressive du prototype.

4.2.2 . Prototypes multiples pour la classe *NULLE*

La seconde méthode, **MultiProto**, cherche à construire plusieurs prototypes pour cette classe *NULLE*. L'idée ici est de partitionner l'espace de sorte à créer plusieurs prototypes pour cette classe. Cette idée s'inspire initialement du travail de Prabhu et al. (2018) dans le domaine de la détection de maladies dermatologiques. Prabhu et al. (2018) ont introduit cette approche pour résoudre deux défis majeurs de leur tâche. Tout d'abord, ils devaient faire face à une distribution de classes à longue queue, avec de nombreuses classes peu fréquentes, impliquant donc d'apprendre à partir de peu d'exemples pour ces classes. De plus, ils rencontraient une grande variabilité entre les exemples au sein d'une même classe, ce qui est similaire aux problèmes que nous rencontrons dans la détection d'événements, en particulier pour la classe *NULLE*. Dans leur travail, ils ont utilisé un réseau prototypique pour mieux traiter les classes peu fréquentes et ont construit plusieurs prototypes pour chaque classe afin de mieux capturer la diversité au sein de ces classes. Cependant, dans notre approche, nous construisons plusieurs prototypes uniquement pour la classe *NULLE*. Cette formulation avec des prototypes multiples a également été adoptée plus récemment pour la classification d'images à partir de peu d'exemples (*Few-Shot Image Classification*) par Huang et al. (2021) et pour la classification de relations par Liu et al. (2022c).

En pratique, nous commençons par la mise en œuvre d'un processus d'apprentissage épisodique standard, similaire à celui que nous avons décrit précédemment dans le chapitre 3. L'objectif de cette phase initiale est de former un encodeur capable de générer des représentations vectorielles pour chaque mot. Ensuite, lors de la phase d'évaluation, nous adoptons une approche épisodique où nous évaluons une phrase de l'ensemble d'évaluation par épisode. Pour résoudre notre problème de détection d'événements,

nous calculons les prototypes des classes d'événements comme précédemment. Cependant, cette fois-ci, nous appliquons un algorithme de clustering aux exemples de la classe *NULLE* présents dans le support set. Cela nous permet de regrouper ces exemples en clusters, chacun représentant une variation de la classe *NULLE*. Une fois que nous avons obtenu ces clusters, nous utilisons la moyenne des vecteurs des mots à l'intérieur de chaque cluster pour former de nouveaux prototypes pour la classe *NULLE*. La détection d'événements est ensuite effectuée en associant une classe à chaque mot en fonction de sa similarité aux prototypes ainsi formés en plus des prototypes de classes d'événements.

Nous avons exploré deux variantes de cette approche. Dans la première variante (ProtoNulle), nous construisons un prototype de la classe *NULLE* pendant la phase de méta-apprentissage, ce qui signifie que nous essayons de regrouper les mots de cette classe même s'ils ne sont pas sémantiquement proches. En revanche, la seconde variante (wo-ProtoNulle) ne construit pas de prototype pour la classe *NULLE* pendant l'apprentissage. Elle se concentre uniquement sur les déclencheurs sans imposer de signal d'apprentissage aux mots de la classe *NULLE*.

Pour le clustering des éléments de la classe *NULLE*, nous adoptons l'algorithme DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) implémenté par Scikit-learn [2]. DBSCAN est un algorithme de clustering largement utilisé en analyse de données et en apprentissage automatique. Sa principale caractéristique réside dans sa capacité à regrouper des données similaires en fonction de leur densité spatiale, sans imposer de forme géométrique particulière aux groupes créés.

Cet algorithme de clustering repose sur deux paramètres clés : epsilon (ϵ) et le nombre minimum de points (`MinPts`). Epsilon représente la distance maximale entre deux points pour qu'ils soient considérés comme voisins et `MinPts` est le nombre minimum de points voisins requis pour qu'un point soit inclus dans un cluster. Le processus de regroupement se déroule en sélectionnant un point de donnée non visité, puis en explorant son voisinage pour identifier d'autres points proches. Si un nombre suffisant de points voisins est trouvé, un cluster est formé. Ce processus se répète pour d'autres points jusqu'à ce que tous les points aient été visités.

Les avantages de DBSCAN résident dans sa capacité à détecter des clusters de formes arbitraires, à gérer les données bruitées et à ne pas exiger la spécification préalable du nombre de clusters. Cependant, il est sensible aux paramètres ϵ et `MinPts`, qui influencent les résultats. De plus, son coût de calcul peut être élevé pour de grands ensembles de données.

Nous donnons une illustration de cette méthode à la figure 4.2. À gauche, nous avons le cas avec un seul prototype pour la classe *NULLE* et à droite, le mécanisme de cluste-

2. https ://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

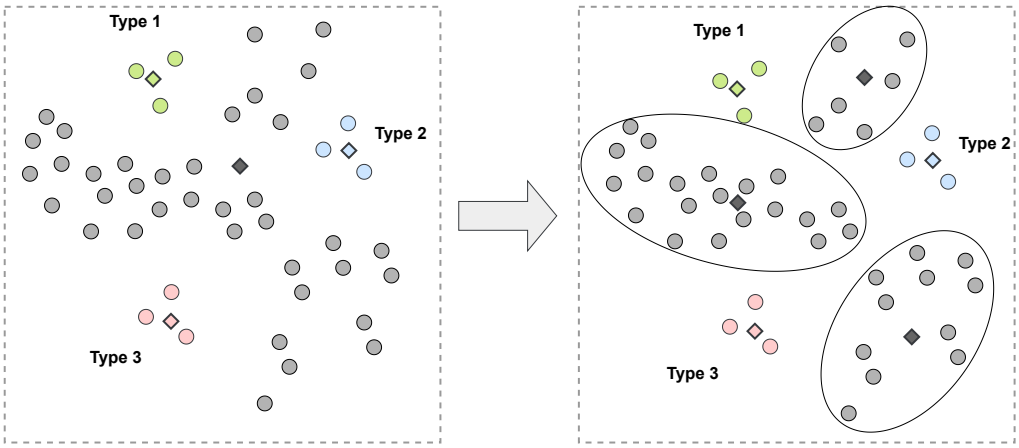


Figure 4.2 – Aperçu du modèle **MultiProto**. Chaque cercle représente un point de donnée et les losanges représentent les prototypes. Chaque couleur représente une classe, le gris étant la classe *NULLE*.

ring crée trois prototypes pour cette classe. Ce schéma illustre l'intérêt que possède la construction de plusieurs prototypes pour la classe *NULLE*, permettant ainsi de mieux couvrir l'ensemble des mots non-déclencheurs.

4.3 . Traitement de la classe *NULLE* par seuillage

La seconde méthode employée consiste à ne pas construire de prototype pour la classe *NULLE* et à calculer un seuil de décision pour filtrer les prédictions. Cette idée a été inspirée par des travaux en TAL sur la classification d'exemples hors domaine avec peu d'exemples (Tan et al., 2019; Nimah et al., 2021).

En effet, les réseaux prototypiques se montrent particulièrement efficaces dans la tâche de classification de textes, dont l'objectif est de classifier globalement une phrase ou un texte en fonction du thème qu'il aborde. Cette efficacité repose particulièrement sur la capacité des modèles de langue contextuels tels que BERT à bien capturer la sémantique des phrases. Ces réseaux ont par exemple été utilisés pour l'analyse de sentiments, la classification de textes ou la classification d'événements qui est également une tâche de classification de phrases [3]. La tâche de classification d'événements comporte également une classe *NULLE* puisqu'il y a des phrases sans aucun événement; mais elle est souvent traitée exactement comme les autres classes (Deng et al., 2020; Lai et al., 2020a; Deng et al., 2021; Lai and Nguyen, 2019; Lai et al., 2021a). Dans cette tâche de classification d'événements, la donnée du déclencheur [4] en plus de la phrase permet d'apporter suffisamment d'informations pour distinguer la classe *NULLE*.

3. En pratique, on cherche à classifier une phrase étant donné un déclencheur.
4. Pour les phrases qui ne comportent aucun événement, le déclencheur est un mot choisi aléatoirement dans la phrase. Lai et al. (2021a) proposent d'utiliser des mots avec les mêmes catégories morpho-syntaxiques (*PoS (Part-of-Speech) tags*) que les vrais déclencheurs pour rendre la tâche plus réaliste.

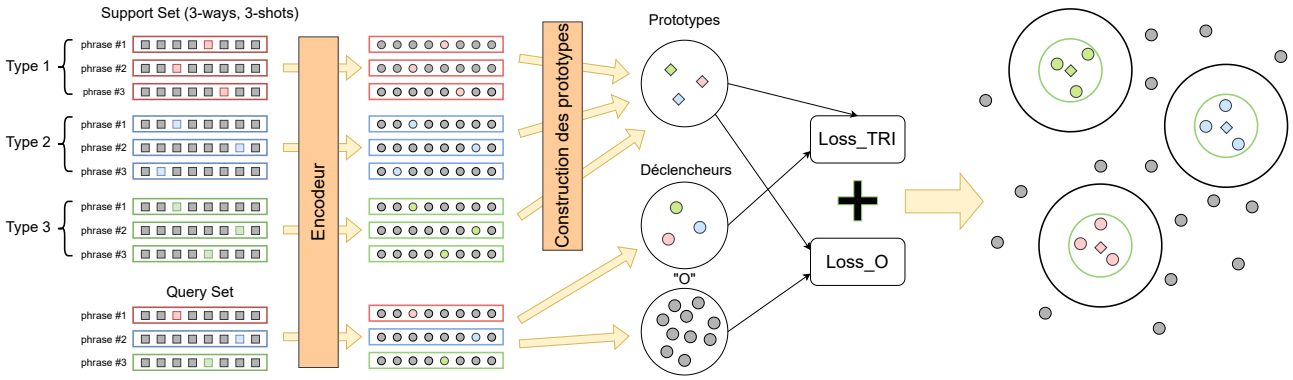


Figure 4.3 – Vue d'ensemble du modèle de détection d'événements à partir de peu d'exemples par apprentissage contrastif et seuillage.

L'article de Tan et al. (2019) est le premier travail à proposer de dédier un traitement spécifique à cette classe *NULLE* dans un contexte d'apprentissage frugal en données. Tan et al. (2019) traitent en particulier la tâche de classification de sentiment sur le jeu de données Amazon Reviews (He and McAuley, 2016) et de classification d'intentions pour un agent conversationnel. Leur objectif est de déterminer si un commentaire sur un article Amazon est positif ou non, mais aussi de pouvoir dire si le commentaire ne porte pas sur l'article en question. De même pour la classification d'intentions, Tan et al. (2019) cherchent à classifier les intentions dans un système d'agent conversationnel ou dire si la requête de l'utilisateur ne fait pas partie des intentions prises en compte par l'agent. Ils proposent pour cela un modèle par apprentissage contrastif qui permet d'écarter les exmples hors domaines des représentations des vraies classes. Leur idée a ensuite été reprise par Nimah et al. (2021), qui ont proposé une nouvelle fonction de coût plus adaptée permettant d'améliorer les performances du modèle.

Nous adaptons cette approche à notre tâche de détection d'événements en considérant la classe *NULLE* comme une classe hors domaine (Schölkopf et al., 2001). Notre objectif est donc de classifier les instances selon les types d'événements connus ou de dire s'ils n'appartiennent à aucune des classes d'événements.

Il faut toutefois noter une différence majeure par rapport aux travaux de Tan et al. (2019). Nous ne faisons pas la classification au niveau de la phrase, mais au niveau du mot et, de plus, les mots de la classe *NULLE* ne sont pas vraiment des exemples hors domaine puisqu'ils font effectivement partie du champ lexical du domaine des événements. Cette subtilité ajoute une couche de complexité à notre problème de détection d'événements.

Une vue d'ensemble du modèle complet est donnée à la figure 4.3. Il est composé des mêmes modules que les modèles précédents à la seule différence qu'il ne construit pas de prototype pour la classe *NULLE*.

Comme précédemment, nous travaillons avec un ensemble composé de N types d'événements, chacun ayant k exemples annotés. Un module d'encodage est utilisé pour traiter ces phrases en entrée et générer une représentation contextuelle pour chaque

mot. Pour une phrase $x = w_1, \ldots, w_L$ de longueur L, l'encodeur produit une séquence $\bar{\mathbf{e}} = \mathbf{e_1}, \ldots, \mathbf{e_L}$, où $\mathbf{e_i}$ est la représentation vectorielle du mot w_i. Par défaut, nous utilisons l'encodeur « *BERT-Weighted* » proposé dans le chapitre précédent et qui a donné les performances les plus élevées pour la tâche considérée.

Ces représentations sont ensuite utilisées pour construire un prototype pour chaque type d'événements en faisant la moyenne des représentations des mots déclencheurs du *support set*. Par la suite, les mots du *query set* sont classifiés en fonction de leur similarité à ces prototypes.

Contrairement à Tuo et al. (2022b) et Cong et al. (2021), nous ne construisons pas de prototype pour la classe *NULLE*. En lieu et place, nous nous appuyons sur un seuil de similarité pour déterminer si un mot est considéré comme déclencheur ou non.

4.3.1 . Processus d'entraînement

Comme nous l'avons souligné dans le chapitre précédent, l'apprentissage dans ce cadre se résume à la modification d'un encodeur capable de générer des plongements similaires pour les déclencheurs appartenant au même type d'événement, tout en maintenant une distance significative entre les déclencheurs et les mots de la classe *NULLE*. Dans cette nouvelle formulation, la similarité entre les mots non-déclencheurs n'est pas une condition nécessaire. Ce qui compte avant tout, c'est d'induire une séparation nette entre les déclencheurs et les mots non-déclencheurs, même si ces derniers peuvent appartenir au même champ lexical.

Modification de la fonction de coût

La fonction de coût habituellement utilisée dans les réseaux prototypiques et dans les problèmes de classification de façon générale est l'entropie croisée (*cross-entropy*). Nous avons adopté ici une fonction de coût de charnière (*hinge-loss)* qui est, de façon générale, plus adaptée à l'apprentissage de similarités ou de dissimilarités. Contrairement à l'entropie croisée, dont l'objectif est d'apprendre à prédire des valeurs de probabilité à partir d'une entrée, les fonctions de charnière prédisent plutôt la similarité relative entre les entrées. Une telle fonction est plus appropriée dans notre cas puisque nous cherchons justement à rendre les déclencheurs plus proches de leurs prototypes que les mots non-déclencheurs. L'expression de l'entropie croisée est rappelée dans l'équation 5.2.

Pour une classe y donnée, la fonction de coût proposée comporte deux termes de fonction de type hinge loss :

- **Loss-TRI** : permet de rapprocher le déclencheur $\mathbf{e_{tr}}$ de son prototype $\mathbf{c^y}$ tout en l'écartant des autres prototypes $\{\mathbf{c^j} \mid j \neq y\}$. Ce terme favorise une séparation nette entre les classes d'événements tout en maintenant la cohérence interne au sein de

chaque classe.

$$\mathcal{L}_{TRI}(\bar{\mathbf{e}}, y) = \sum_{j \neq y} \max\left(0, \mathcal{M}_0 + s\left(\mathbf{e_{tr}}, \mathbf{c^j}\right) - s\left(\mathbf{e_{tr}}, \mathbf{c^y}\right)\right) \tag{4.1}$$

- **Loss-O** : permet quant à lui d'écarter les mots de la classe « O » $\mathbf{e_i}\,(i \neq tr)$ de tous les prototypes c^j. Ce terme permet de garantir la séparabilité entre les mots non-déclencheurs et les classes d'événements.

$$\mathcal{L}_O(\bar{\mathbf{e}}) = \max_{i \neq tr}\left(0, \max_j\left(s\left(\mathbf{e_i}, \mathbf{c^j}\right) - \mathcal{M}_1\right)\right) \tag{4.2}$$

où $s(.)$ est une fonction de similarité, $\mathcal{M}_0$ et $\mathcal{M}_1$ sont des hyper-paramètres correspondant à des marges. $\mathcal{M}_0$ permet de contrôler la distance de l'exemple positif le plus éloigné de son prototype. En d'autres termes, il permet de contrôler le degré de compacité des exemples au sein d'une même classe. Un compromis està trouver pour cet hyper-paramètre : une classe trop compacte peut mal prédire les déclencheurs rares, tandis qu'une classe trop dispersée peut entraîner des confusions avec d'autres classes. À l'inverse, $\mathcal{M}_1$ contrôle la distance du mot non-déclencheur le plus proche. La différence entre $\mathcal{M}_1$ et $\mathcal{M}_0$ est la largeur de la marge souhaitée entre les exemples positifs et les exemples négatifs. Chacun des deux termes ci-dessus peut par ailleurs être remplacé par n'importe quel autre type de fonction de coût de façon complètement indépendante. Par exemple, le travail de Tan et al. (2019) avait utilisé une fonction de coût de type *cross-entropy* pour le premier terme et proposé un troisième terme (*hinge-loss*) pour contrôler les écarts entre les prototypes. Nous n'avons pas trouvé utile d'utiliser ce troisième terme car il était redondant avec le premier dans notre cas et n'apportait pas de gain significatif.

Le modèle est entraîné en minimisant une fonction de coût globale égale à la somme de ces deux termes.

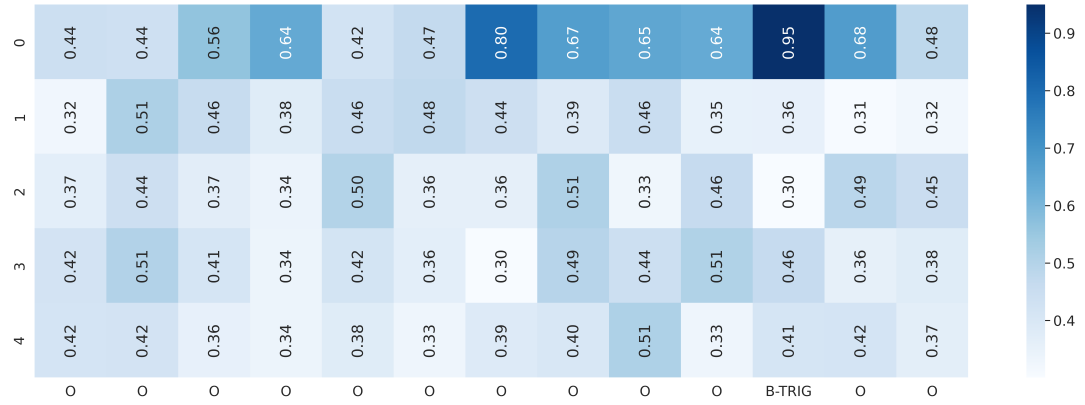
4.3.2 . Module de prédiction

L'approche standard avec les réseaux prototypiques est de classer chaque exemple en fonction de sa similarité avec les prototypes. Dans notre modèle, en l'absence de prototype pour la classe *NULLE*, nous devons nous fier à un seuil en dessous duquel le mot est considéré comme un non-déclencheur.

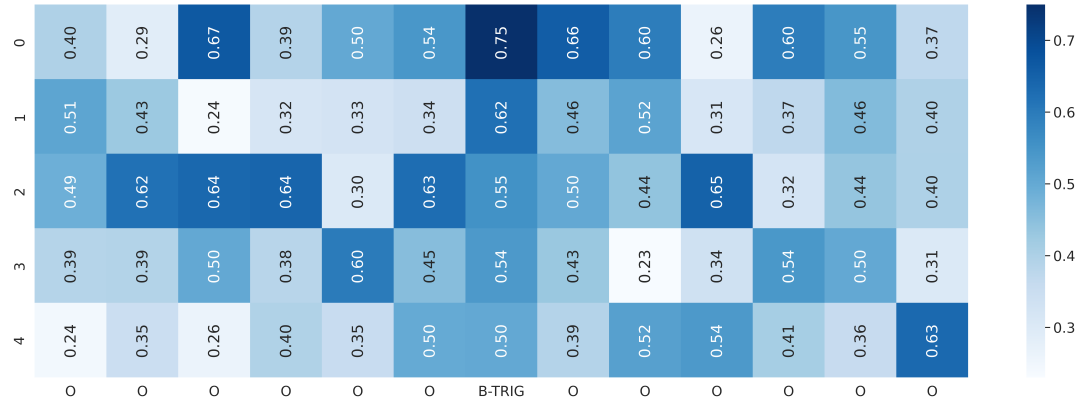
Dans les travaux de Tan et al. (2019) et Nimah et al. (2021), un seuil optimal est défini en utilisant la distribution des valeurs de similarité sur un ensemble de validation. Le seuil ainsi déterminé est ensuite appliqué aux exemples de l'ensemble d'évaluation.

Cependant, dans notre cas, nous avons constaté de manière empirique que les distributions des valeurs de similarité entre un déclencheur et les prototypes varient considérablement d'une phrase à l'autre. Cela rend l'utilisation d'un seuil global impraticable. À

titre d'exemple, dans la figure 4.4, nous présentons deux exemples de la classe `Justice.Arrest-Jail` [5] dans une configuration avec cinq classes par épisode. On peut observer que pour la phrase de la figure 4.4a, un seuil optimal serait situé entre 0,95 et 0,80 tandis que pour la phrase de la figure 4.4b, le seuil optimal devrait être placé entre les valeurs 0,75 et 0,67. Cette disparité implique qu'il n'est pas possible de définir un seul seuil qui fonctionnerait de manière cohérente dans tous les cas, ce qui rend cette approche inadaptée. Une approche plus flexible et adaptative est de ce fait nécessaire pour gérer cette variabilité inhérente aux données.



(a)



(b)

Figure 4.4 – Valeurs de similarité pour deux phrases différentes appartenant à la même classe dans une configuration avec cinq prototypes. Les lignes correspondent aux prototypes et les colonnes aux étiquettes des mots. `B-TRIG` indique la position du déclencheur.

Cette observation souligne l'importance de la variabilité contextuelle : même des phrases partageant des déclencheurs identiques peuvent présenter des distributions de similarité différentes. Cette variabilité devient encore plus prononcée lorsque l'on considère d'autres événements ou d'autres déclencheurs, où les contextes et les expressions linguistiques peuvent varier de manière significative.

5. Correspondant à l'arrestation et l'emprisonnement d'une personne.

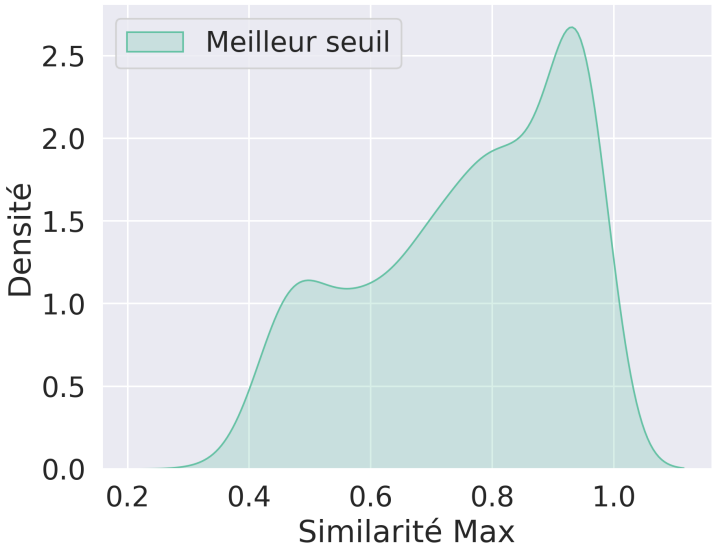


Figure 4.5 – Distribution des seuils optimaux sur l'ensemble de validation.

Nous avons représenté la distribution des seuils optimaux sur l'ensemble de validation à la figure 4.5. Nous constatons que ces seuils varient considérablement, bien qu'il y ait une concentration notable vers les valeurs élevées (ce qui correspond au comportement souhaité). Cette observation confirme que l'application d'un seuil global pour tous les exemples ne serait pas appropriée. Par conséquent, nous proposons une approche qui permet de déterminer dynamiquement un seuil adapté à chaque phrase, tenant ainsi compte de la variabilité inhérente aux données et améliorant la précision des prédictions dans des contextes diversifiés.

4.3.3 . Recherche d'un seuil dynamique

Notre approche consiste à déterminer la probabilité associée au seuil optimal en utilisant la fonction de répartition sur les valeurs maximales de similarité. Cette méthode nous permet d'ajuster dynamiquement le seuil en fonction des caractéristiques de chaque exemple, offrant ainsi une meilleure capacité à distinguer les déclencheurs des mots non-déclencheurs dans des contextes variés.

Plus précisément, étant donné que les similarités entre les déclencheurs et les prototypes sont plus élevées que celles des mots « O » de façon générale, nous supposons que, pour une phrase donnée, les similarités des déclencheurs n'apparaîtront qu'au-dessus d'un certain quantile assez élevé dans la distribution des similarités des mots de la phrase. Nous supposons également que ce quantile est suffisamment stable, même s'il ne correspond pas exactement à la même valeur de similarité d'un exemple à l'autre. Il suffit donc de déterminer ce quantile pour pouvoir déterminer le seuil adapté à chaque phrase.

En pratique, pour une phrase donnée du *query set*, nous sélectionnons une phrase témoin qui sera utilisée pour déterminer la probabilité du seuil optimal en deux étapes.

- **Recherche de la phrase témoin :** nous cherchons dans le *support set* la phrase la plus similaire à la phrase considérée. Dans ce contexte, la similarité entre deux phrases est calculée comme la moyenne des similarités entre tous les mots composant ces deux phrases (y compris le jeton `[CLS]` du modèle BERT).
- **Recherche du seuil de similarité :** nous calculons ensuite les similarités entre la phrase témoin et les prototypes, faisons varier le seuil entre les similarités minimum et maximum et adoptons finalement celui maximisant la F1-mesure sur la phrase témoin sélectionnée. Ensuite, nous déterminons la probabilité correspondant à ce seuil en utilisant la fonction de répartition sur ces valeurs de similarité. Enfin, nous déterminons le seuil optimal pour la phrase du *query set* à partir de sa fonction de répartition et de la probabilité déterminée précédemment.

Toutefois, comme les probabilités empiriques directement calculées à partir de la fonction de répartition dépendent du nombre de mots dans les phrases, nous interpolons linéairement la fonction de répartition sur un plus grand nombre de points avant d'estimer les probabilités, ce qui nous permet de donner artificiellement à toutes les phrases la même longueur. Nous utilisons 512 points pour faire cette interpolation car cela correspond au nombre maximal de mots pouvant être pris en entrée du modèle BERT, qui est notre encodeur par défaut.

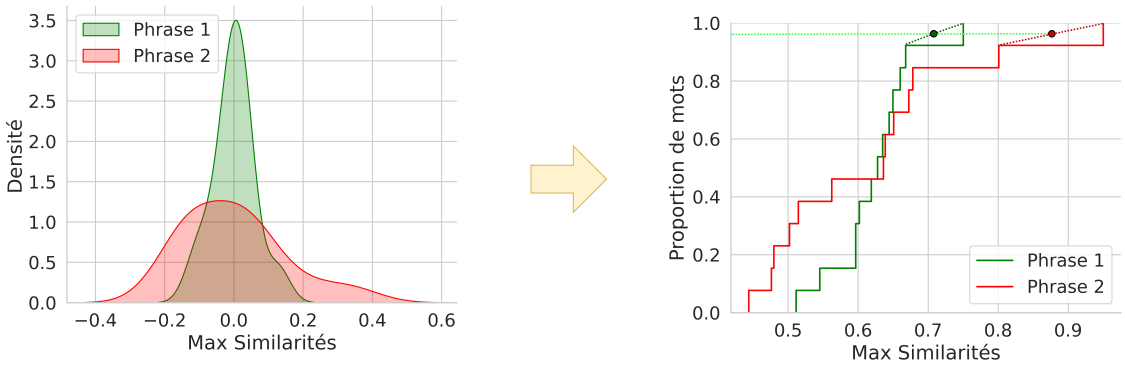
Nous donnons une illustration de la recherche du seuil dynamique par la fonction de répartition à la figure 4.6. Dans cet exemple, la « phrase 1 » (phrase témoin du *support set*) a un seuil optimal estimé à $0{,}71$ correspondant à une probabilité estimée à $P = 0{,}97$. Nous reportons ensuite cette valeur de probabilité sur la fonction de répartition de la « phrase 2 » (phrase du *query set* à classifier) pour obtenir son seuil optimal, égal à $0{,}92$.

Nous répétons ce même processus pour tous les exemples de l'ensemble d'évaluation, permettant ainsi de déterminer un seuil adapté à chaque exemple.

Filtrage par les étiquettes morphosyntaxiques

Au cours de nos expériences préliminaires et des travaux présentés dans le chapitre 3, nous avons constaté un déséquilibre entre la précision ($\approx 65\%$) et le rappel ($\approx 80\%$) de notre modèle de détection d'événements. Ce déséquilibre suggère une présence notable de faux positifs.

Pour surmonter ce problème, nous avons introduit une étape de filtrage supplémentaire qui s'appuie sur les catégories morphosyntaxiques des mots. Nous ne conservons ainsi que les prédictions associées aux étiquettes morphosyntaxiques les plus couramment liées aux déclencheurs d'événements, tels que les noms et les verbes. Cette approche vise à réduire les faux positifs, améliorant ainsi la précision globale de notre modèle.



(a) **(b)**

Figure 4.6 – Illustration de la méthode de seuillage dynamique à l'aide la fonction de répartition. **(a)** donnent les histogrammes des similarités et **(b)** les fonctions de répartition.

Ce filtrage par catégories morphosyntaxiques a un impact positif sur les performances globales du modèle. Il contribue à augmenter la précision en réduisant le nombre de faux positifs. Cependant, cette amélioration de la précision s'accompagne d'une légère baisse du rappel. Au total, ces ajustements se traduisent par une augmentation d'environ 5 points de pourcentage sur la mesure F1-mesure en moyenne. Nous présentons les résultats détaillés dans le tableau 4.5.

4.4 . Expérimentations

4.4.1 . Paramètres expérimentaux

Jeux de données

Nous avons mené nos expérimentations sur les jeux de données ACE-2005 (Walker et al., 2006), MAVEN (Wang et al., 2020b) et FewEvent (Deng et al., 2020).

ACE-2005 est le jeu de données issu de la campagne d'évaluation éponyme. Il est composé de 8 types d'événements subdivisés en 33 sous-types. Nous adoptons le découpage proposé par Lai et al. (2020a), qui prend quatre des types pour l'ensemble d'entraînement et les quatre autres pour l'ensemble d'évaluation. On peut voir ces huit types d'événements comme étant issus de domaines différents qui sont *Life, Transport, Justice, Conflict, Contact, Business, Personnel* et *Transaction*. L'objectif ici est de former des ensembles d'apprentissage et d'évaluation n'étant pas du même domaine. Nous donnons la liste des types et sous-types dans le tableau A.1 en Annexe A. Nous avons traité les données ACE-2005 de sorte à ne garder que les phrases contenant au moins une mention d'événement.

MAVEN est un jeu de données conçu uniquement pour la tâche de détection d'événements (non annoté en entités) tout en étant bien plus riche que le jeu de données ACE-2005. Il comporte 168 sous-types d'événements répartis en 33 types principaux (qui

incluent les types de ACE-2005). Nous avons utilisé le découpage de Chen et al. (2021b) comportant 120 types dans l'ensemble d'entraînement et 45 dans l'ensemble d'évaluation.

FewEvent est l'ensemble de données conçu spécialement pour la détection d'événements à partir de peu d'exemples, utilisé dans le chapitre précédent.

Dans tous les cas, les ensembles d'évaluation et d'apprentissage contiennent des classes distinctes, de sorte que, lors de l'évaluation, le modèle soit confronté à de nouvelles classes non vues pendant l'entraînement.

Pendant l'entraînement du modèle, nous ne considérons qu'un seul événement par phrase afin de respecter les conditions N-ways, k-shots. Mais pendant l'évaluation, les phrases sont traitées sans a priori et peuvent contenir plus d'un événement.

Méthode d'évaluation et hyper-paramètres

Nous adoptons l'évaluation épisodique N-ways, k-shots, qui consiste à construire des épisodes avec N classes et k exemples annotés par classe. Dans l'évaluation épisodique standard (Vinyals et al., 2016), les ensembles de test sont échantillonnés de façon à ce que toutes les classes soient distribuées uniformément, ce qui ne correspond pas à la distribution des mentions d'événements dans les données réelles. Ainsi, les scores de performance rapportés par cette méthode ne reflètent pas la distribution réelle des données. Nous adoptons la configuration plus réaliste de Yang and Katiyar (2020), qui construit le *support set* avec $N \times k$ exemples et évalue le modèle sur tous les autres exemples en traitant un exemple de l'ensemble d'évaluation par épisode jusqu'à épuiser l'ensemble d'évaluation. Dans ce cas, le nombre d'épisodes d'évaluation correspond au nombre d'exemples de test sans les $N \times k$ exemples pris pour constituer le support set.

Pour les expériences, nous avons utilisé le modèle pré-entraîné BERT (Devlin et al., 2019a) comme encodeur et adopté la stratégie *Weighted* pour obtenir des représentations contextuelles des mots. Nous adoptons une longueur maximale de séquence de 128 tokens, un taux d'apprentissage de $1e-5$ et 30 000 épisodes N-ways, k-shots pour entraîner le modèle. Les hyper-paramètres $\mathcal{M}_0 = 1$ et $\mathcal{M}_1 = 0{,}4$ ont été obtenus sur l'ensemble de validation, pris entre $0{,}2$ et 1 (avec un pas de $0{,}2$). Pour l'algorithme de clustering DBSCAN, nous fixons $\texttt{MinPts} = k$ pour garantir qu'il y ait au moins k exemples par cluster formé et $\epsilon = 0,6$ est déterminé sur l'ensemble de validation.

4.4.2 . Comparaison des approches proposées

Nous présentons dans cette section une comparaison des différentes approches que nous avons proposées pour le traitement de la classe NULLE, en évaluant leurs performances sur le jeu de données FewEvent.

Performances pour les méthodes par redéfinition du prototype

Nous présentons dans un premier temps les performances pour les méthodes par redéfinition du prototype de la classe *NULLE* dans le tableau 4.2. Nous reportons ces résultats sur le jeu de données FewEvent dans une configuration 5-ways, 5-shots. Le modèle entier est un modèle Proto-dot décrit dans le chapitre 3 et l'encodeur utilisé est un encodeur **BERT-Weighted**.

Méthode		**Précision**	**Rappel**	**F1-mesure**
StableProto	Proto-Zero	11,64	67,48	19,85 ($\downarrow 45,95$)
	Proto-Learn	26,59	70,62	38,63 ($\downarrow 27,17$)
MultiProto	ProtoNulle	62,72	72,32	67,18 ($\uparrow 1,37$)
	wo-ProtoNulle	65,35	72,34	68,66 ($\uparrow 2,85$)

Table 4.2 – Performances pour les méthodes par redéfinition du prototype de la classe *NULLE*. ($\downarrow$) et ($\uparrow$) caractérisent les variations de performance par rapport au modèle Proto-dot de référence qui construit un prototype pour la classe *NULLE*.

Nous observons que les méthodes **StableProto** ne parviennent pas à atteindre les mêmes niveaux de performance que le modèle de référence Proto-dot. Ces approches ont du mal à saisir la sémantique variée de la classe *NULLE*. En revanche, les méthodes qui visent à construire plusieurs prototypes pour la classe *NULLE* présentent un intérêt certain : elles contribuent en particulier à accroître la précision, ce qui se traduit par une amélioration des performances en termes de F1-mesure. Cependant, ces méthodes ont tendance à réduire le rappel car la multiplication des prototypes pour la classe *NULLE* accorde une plus grande importance à cette classe dans l'espace des caractéristiques, ce qui entraîne une légère augmentation du nombre de faux négatifs.

Ces observations confirment également la difficulté d'apprendre un ou plusieurs prototypes pour la classe *NULLE* et donc l'intérêt de méthodes par seuillage.

Performances des méthodes par seuillage

Nous présentons ici les résultats pour les méthodes par seuillage dynamique. Pour démontrer l'intérêt de la méthode que nous proposons, nous la comparons à plusieurs autres façons de déterminer le seuil de façon dynamique (voir tableau 4.3). Nous avons délibérément omis les méthodes visant à déterminer un seuil global car elles se sont avérées bien moins performantes. Pour preuve, nous traçons à la figure 4.7 la précision, le rappel et la F1-mesure en faisant varier le seuil entre 0 et 1 sur le jeu de données FewEvent, dans le cadre de la condition 5-ways, 5-shots. Ces résultats démontrent que la F1-mesure optimale obtenue avec un seuil global plafonnerait à environ $0{,}47$. Cette observation met en évidence l'avantage manifeste de privilégier un seuillage dynamique plutôt qu'un seuil global adopté dans des travaux tels que ceux de Tan et al. (2019) et Nimah et al. (2021).

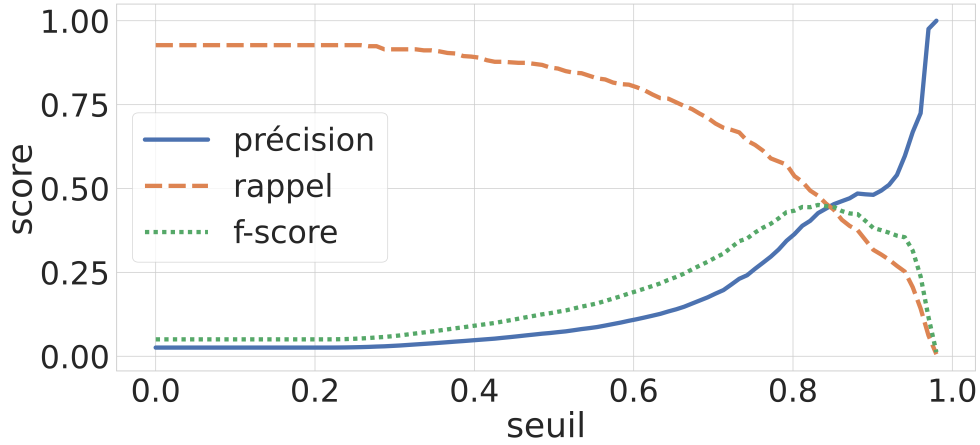


Figure 4.7 – Variation des performances avec un seuil global sur le jeu de données FewEvent.

Nous nous focalisons ici sur les méthodes qui utilisent un exemple témoin pour la recherche du seuil. Cette approche a été testée en utilisant soit le support set d'un épisode, soit à l'aide d'un exemple de l'ensemble de validation. Globalement, les méthodes utilisant le support set se sont montrées un peu meilleures que les méthodes utilisant l'ensemble de validation. Cette observation peut résulter de ce que dans le support set d'un épisode, nous avons des exemples de la même classe que celui que nous cherchons à classifier, et même parfois des exemples contenant exactement le même déclencheur, mais dans un autre contexte alors que les classes de l'ensemble de validation sont disjointes des classes d'évaluation. De plus, la recherche d'un exemple témoin est plus coûteuse sur l'ensemble de validation car elle demande de considérer plus de phrases alors qu'il n'y a que $N \times k$ phrases dans le support set. L'utilisation de l'ensemble de validation conduit ainsi à une inférence trop longue.

En résumé, nous faisons l'évaluation du modèle phrase par phrase, chaque phrase étant traitée au cours d'un épisode. L'ensemble de support (support set) reste inchangé pour une classe donnée et les exemples du support set ne sont pas évalués. Le query set n'est composé que de l'exemple à traiter, que nous appellerons phrase query dans la suite. Au cours d'un épisode, nous sélectionnons une phrase similaire à la phrase query, appelée phrase témoin. Nous adoptons une évaluation N-ways, k-shots, c'est-à-dire avec N classes par épisode et k exemples par classe dans le support set. En pratique, pour former un épisode, nous prenons la classe de l'exemple query, puis $N - 1$ autres classes parmi les classes de l'ensemble d'évaluation, ces classes pouvant changer d'un épisode à l'autre.

Nous comparons cinq façons de déterminer le seuil de façon dynamique et présentons les résultats dans le tableau 4.3 :

- **Threshold** retient le seuil directement estimé sur l'exemple témoin, seuil que l'on reporte sur l'exemple query. Cette méthode, qui consiste à directement transférer le seuil obtenu d'une phrase à une autre, est la plus naturelle.
- **Max-O** consiste à prendre comme seuil le maximum des valeurs de similarité pour les mots non-déclencheurs dans la phrase témoin. L'idée sous-jacente à cette méthode est de considérer que si l'apprentissage se fait correctement, les valeurs de similarité pour les mots non-déclencheurs devraient toutes être en dessous de celles des déclencheurs. Donc la valeur maximale des similarités sur les non-déclencheurs peut constituer un bon seuil.
- **Min-Trigger** consiste à l'inverse à prendre le minimum des valeurs de similarité des déclencheurs dans le support set. Cette méthode est motivée par une raison similaire à la méthode précédente. Nous faisons l'hypothèse que, normalement, le seuil optimal devrait se situer juste en dessous des valeurs de similarité des déclencheurs, donc la valeur minimale sur les déclencheurs pourrait être un bon seuil candidat.
- **Max-Epsilon** correspond au fait de prendre la valeur maximale sur les similarités puis à y retirer une petite valeur ϵ pour trouver le seuil. En général, il n'y a qu'un seul événement par phrase et la similarité correspondant au déclencheur est la valeur maximale. Nous prenons donc un seuil juste en dessous de cette valeur maximale. Nous avons testé avec la valeur de $\epsilon = 0,05$.
- **Centered** consiste à centrer et réduire les distributions des valeurs de similarité, puis à appliquer le même seuil aux deux phrases après cette opération. Le processus de centrage et de réduction est effectué en soustrayant la moyenne des valeurs de similarité et en divisant par l'écart-type, ce qui permet de standardiser les distributions. La valeur du seuil sur la phrase témoin, T_1, subit la modification suivante $T_1^{'} = \frac{T_1 - \mu_1}{\sigma_1}$ et la valeur du seuil sur la phrase query est obtenue par :

  $$T_2 = \frac{\sigma_2}{\sigma_1}(T_1 - \mu_1) + \mu_2$$

  où μ_1, μ_2, σ_1 et σ_2 sont respectivement les moyennes et écarts-types des similarités sur la phrase témoin et la phrase query.
  Nous donnons une illustration de cette méthode à la figure 4.8.
- **ECDF** utilise la fonction de répartition pour estimer le seuil. C'est la méthode décrite en détail précédemment.
- **Best-Threshold** est enfin le seuil calculé directement sur l'exemple à classifier. Ce seuil « oracle » nous donne une indication du meilleur résultat pouvant être obtenu avec notre formulation du problème.

Les méthodes **Min-Trigger**, **Max-O** et **Threshold** ont tendance à fixer des seuils trop variables pour fonctionner sur lesexemples d'évaluation. Ces méthodes sont, par défini-

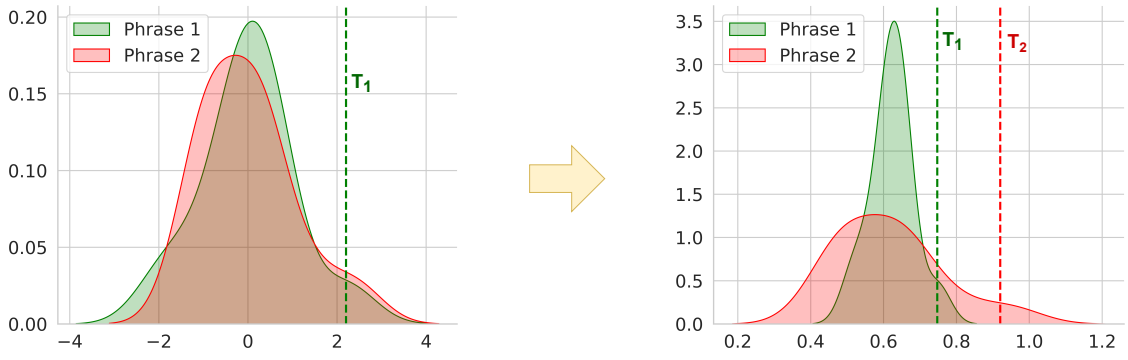


Figure 4.8 – Illustration de la méthode **Centered**.

Méthode		**Précision**	**Rappel**	**F1-mesure**
Validation	Threshold	40,32	48,14	44,52 ($\downarrow 21,28$)
	Min-Trigger	32,54	24,41	27,44 ($\downarrow 38,37$)
	Max-O	43,78	21,82	29,13 ($\downarrow 38,68$)
	Max-Epsilon	54,87	48,23	51,33 ($\downarrow 14,47$)
	Centered	62,41	76,34	68,67 ($\uparrow 2,86$)
	ECDF	66,66	81,11	73,18 ($\uparrow 7,37$)
Support set	Threshold	51,40	56,65	53,90 ($\downarrow 11,91$)
	Min-Trigger	33,12	45,32	38,22 ($\downarrow 27,59$)
	Max-O	45,78	25,23	32,53 ($\downarrow 33,28$)
	Max-Epsilon	60,75	55,31	57,90 ($\downarrow 7,91$)
	Centered	62,97	81,88	71,19 ($\uparrow 5,38$)
	ECDF	68,71	82,97	75,17 ($\uparrow 9,36$)
Best-Threshold		81,62	83,97	82,59 ($\uparrow 16,78$)

Table 4.3 – Comparaison des méthodes par seuillage dynamique.

tion, sujettes au défi de la recherche de seuil que nous avons mentionné précédemment. En effet, comme nous l'avons souligné à plusieurs reprises, la fixation d'un seuil issu d'un autre exemple ne fonctionne pas de manière fiable, même si cet autre exemple est très similaire d'un point de vue sémantique à l'exemple query. Cela est très bien illustré avec les deux phrases considérées à la figure 4.4.

La méthode **Max-Epsilon** se détache des trois méthodes précédentes mais reste dépendante de l'écart qu'il peut y avoir entre la valeur maximale et la valeur immédiatement inférieure. De plus, dans le cas où la valeur maximale ne correspond pas au déclencheur, elle a tendance à filtrer ce dernier, augmentant ainsi le nombre de faux négatifs dans de telles situations. Par exemple, dans les cas où le déclencheur a une

valeur de similarité inférieure à celle d'un mot « O », nous préférerions que le seuil soit en dessous de sa valeur de similarité pour garder un faux positif et un vrai positif plutôt que de ne garder qu'un faux positif.

Les deux autres méthodes, **Centered** et **ECDF**, qui visent à estimer la probabilité du seuil optimal, s'avèrent nettement plus performantes. Néanmoins, malgré le gain de performance notable avec ces méthodes, on peut constater que les précisions restent tout de même bien en dessous des rappels, indiquant qu'il y a encore beaucoup de faux positifs prédits. Ces deux méthodes de recherche du seuil sont assez proches en première approximation mais la méthode **Centered** reste tout de même sensible à la variabilité des valeurs de similarité, qui peuvent changer considérablement d'une phrase à l'autre (comme à la figure 4.8).

En pratique, la version **ECDF** considère la même aire sous la courbe pour les deux densités de probabilité, ce qui n'est pas le cas pour la version **Centered**.

Par ailleurs, l'écart important entre les performances obtenues avec le seuil oracle (**Best-Threshold**) et celles résultant des méthodes de seuillage présentées suggère que notre approche pourrait encore être améliorée en trouvant une meilleure façon de déterminer le seuil.

En résumé, ces expérimentations ont mis en évidence une nette supériorité des méthodes par seuillage dynamique par rapport aux méthodes par redéfinition du prototype de la classe *NULLE*. Toutefois, il convient de noter que ces méthodes par seuillage ont été explorées plus en profondeur que les autres méthodes, principalement parce qu'elles avaient donné des résultats plus prometteurs lors des phases préliminaires de l'étude.

4.4.3 . Comparaison avec l'état de l'art

Nous comparons notre approche **ECDF** à d'autres modèles de l'état de l'art dans la configuration 5-ways, 5-shots dans le tableau 4.4. Le modèle **PA-CRF** (Cong et al., 2021) est un modèle qui calcule un prototype pour la classe *NULLE* et estime les probabilités de transition entre les étiquettes `B-I-O` avec l'utilisation d'une couche CRF (*Conditional Random Fields)* (Lafferty et al., 2001). (Tuo et al., 2022b) est notre modèle du chapitre 3.

HCL-TAT (Zhang et al., 2022c) est également un modèle sans prototype pour la classe *NULLE* utilisant un seuil de décision égal à la moyenne des similarités sur l'exemple traité pendant un épisode. Ce travail aborde exactement la même problématique que nous et adopte une méthode similaire qui s'appuie sur un apprentissage contrastif. La principale différence avec notre approche est la fonction de coût utilisée et la méthode de calcul du seuil dynamique.

Nous comparons toutes ces méthodes à un modèle prototypique standard qui construit un prototype pour la classe nulle, utilise l'entropie croisée comme fonction de coût et un encodeur BERT-base (**PROTO**).

Enfin, **FS-Causal** (Chen et al., 2021b) est un modèle ajoutant une prise en compte explicite des relations de causalité entre les déclencheurs et leur contexte pour résoudre le problème dit de la malédiction des déclencheurs (*trigger curse problem*) décrit dans Wang et al. (2021a). En effet, la disparité entre les déclencheurs au sein d'un type d'événement et la rareté de certains déclencheurs conduit souvent à un sur-apprentissage qui nuit à la détection des déclencheurs rares dans l'ensemble de données. La prise en compte du contexte seul (sans les déclencheurs) permet de réduire cet effet de sur-apprentissage. Leur approche consiste donc à associer de façon équilibrée les informations du contexte et celles des déclencheurs pour améliorer les prototypes des classes et réduire ce biais. Comme leurs résultats rapportés ne sont évalués que classe par classe, cela correspond à une configuration 1-way, k-shots (une classe par épisode et k exemples annotés dans le support set). Nous nous mettons dans les mêmes conditions pour comparer notre approche avec leurs résultats.

	Modèle	ACE-2005	MAVEN	FewEvent
5-ways, 5-shots	PROTO	49,2 ± 1,2	51,6 ± 1,4	53,6 ± 0,7
	PA-CRF (Cong et al., 2021)	64,0 ± 0,6	65,2 ± 0,3	65,3 ± 2,0
	(Tuo et al., 2022b)	66,4 ± 1,8	67,1 ± 1,5	67,4 ± 1,1
	HCL-TAT† (Zhang et al., 2022c)	–	–	66,9 ± 0,7
	Notre modèle	74,0 ± 1,1	76,9 ± 1,1	79,6 ± 4,2
1W,5S	FS-Causal† (Chen et al., 2021b)	76,9 ± 1,4	55,0 ± 0,4	–
	Notre modèle	80,9 ± 2,9	81,1 ± 1,1	79,1 ± 2,1

Table 4.4 – Performance de détection d'événement sur trois jeux de données : moyenne et écart-type de la micro F1-mesure sur 5 essais. † indique les résultats issus de l'article original.

Selon le tableau 4.4, notre méthode établit une nouvelle référence de l'état de l'art avec une augmentation importante de la F1-mesure pour les trois jeux de données considérés.

Dans la configuration 1-way 5-shots, notre modèle améliore également les performances par rapport à **FS-Causal** (Chen et al., 2021b) pour les deux jeux de données sur lesquels les évaluations sont réalisées. Ce résultat montre d'abord que l'amélioration apportée par notre proposition n'est pas limitée à un unique cadre d'évaluation. Par ailleurs, considérer de nouveaux types d'événements un par un est la stratégie la plus générale pour l'adaptation à un nouveau domaine dans lequel le nombre de types d'événements n'est pas connu à l'avance.

4.4.4 . Discussions

Nous analysons dans le tableau 4.5 l'impact de chaque composant du modèle au travers d'une étude d'ablation. La ligne – Hinge loss correspond au fait de remplacer

la fonction de coût de charnière par une entropie croisée ; et la ligne – BERT-weighted correspond au fait d'utiliser l'encodeur BERT standard à la place de l'encodeur **BERT-weighted**.

Modèle complet	**79,6 ± 4,2**
– PoS-tags	77,9 ± 3,9 ($\downarrow 1,7$)
– Hinge loss	75,9 ± 5,4 ($\downarrow 2$)
– BERT-weighted	70,9 ± 2,7 ($\downarrow 5$)
– BERT-weighted	74,7 ± 3,0 ($\downarrow 3,2$)

Table 4.5 – Étude d'ablation pour chaque composante du modèle. Moyenne et écart-type de la F1-mesure sur cinq essais dans une configuration 5-ways, 5-shots sur le jeu de données FewEvent. (↓) indique la perte de performance moyenne liée à chaque composante en points de pourcentage. Le niveau d'indentation caractérise les effets cumulés de chaque composante.

Ces analyses suggèrent que l'encodeur **BERT-weighted** et l'apprentissage contrastif avec une fonction de coût de charnière, combinés à notre nouvelle formulation sans prototype de la classe *NULLE*, jouent un rôle important dans la performance globale du modèle. Plus spécifiquement, nous pouvons noter que l'apprentissage contrastif contribue fortement à diminuer la variance des résultats en plus du gain de performance. Cette fonction de coût, combinée à notre stratégie de recherche de seuil, contribue à la forte différence de performance avec **HCL-TAT** (Zhang et al., 2022c), observée dans le tableau 4.4, alors que nos problématiques sont initialement similaires.

Comme nos expériences préliminaires l'ont suggéré, le filtrage des déclencheurs candidats en fonction de leur catégorie morphosyntaxique permet d'augmenter les performances de quelques points de pourcentage pour deux jeux de données. Toutefois, le modèle sans ce filtrage, qui est la plus directement comparable aux modèles de l'état de l'art, montre que celui-ci n'est pas le facteur principal des améliorations obtenues.

Nous constatons un impact significatif de l'encodeur **BERT-weighted**, ce qui souligne l'efficacité de cette méthode pour enrichir les plongements lexicaux dans le contexte de la détection d'événements à partir de peu d'exemples. Cette constatation suggère également que les avantages de cette approche peuvent s'additionner à d'autres améliorations intégrées dans le réseau prototypique global, comme nous l'avions déjà observé dans le chapitre précédent. En effet, les travaux présentés dans ce chapitre apportent une contribution significative, se focalisant principalement sur le module de prédiction au sein du réseau prototypique (comportant à la fois des modifications de la fonction de coût et du processus de prédiction de la figure 3.2).

De façon complémentaire, nous analysons l'impact de chaque terme de la fonction de coût dans le tableau 4.6. Nous comparons les fonctions de coût de type charnière et de type entropie croisée en évaluant spécifiquement l'influence du second terme dans chaque cas. Comme évoqué précédemment, la fonction de coût de type charnière joue

un rôle central dans l'ensemble du modèle. Cela se traduit par une amélioration notable d'environ 6 points de F1-mesure par rapport à la fonction de coût de type entropie croisée.

$\mathcal{L}_{ID}$ = Hinge loss	**77,9 ± 3,9**
$-\mathcal{L}_O$	76,2 ± 3,1 ($\downarrow 1,7$)
$\mathcal{L}_{ID}$ = CE loss	75,9 ± 5,4
$-\mathcal{L}_O$	73,8 ± 4,8 ($\downarrow 2,1$)

Table 4.6 – Ablation pour chaque terme de la fonction de coût. Ces résultats sont donnés sans le filtrage par les catégories morphosyntaxiques. ($\downarrow$) indique la perte de performance liée au second terme dans chaque cas.

Concernant l'impact du second terme, bien que son influence soit positive, elle reste relativement marginale dans les deux scénarios. En effet, ce second terme a pour unique objectif d'éloigner les mots de la classe *NULLE* des prototypes des types d'événements. Cependant, étant donné que les types d'événements diffèrent entre l'ensemble d'entraînement et l'ensemble d'évaluation, l'efficacité de ce terme peut être limitée, voire contre-productive, pour les classes d'évaluation. Il est par exemple possible que lors de l'apprentissage, les exemples de la classe *NULLE* soient involontairement attirés vers les prototypes des classes d'évaluation. Cette situation peut se produire étant donné que les mots de la classe *NULLE* sont généralement communs aux exemples d'apprentissage et d'évaluation, contrairement aux déclencheurs qui sont, en général, spécifiques à chaque type et donc à chaque ensemble.

4.5 . Conclusion

Dans ce chapitre, nous avons abordé le défi posé par la classe *NULLE* dans la tâche de détection d'événements à partir de peu d'exemples à l'aide des réseaux prototypiques. Cette classe, par sa nature hétérogène, représente souvent un obstacle majeur dans la tâche de détection d'événements et nécessite donc un traitement particulier.

Notre démarche s'est articulée autour de deux principales approches. D'une part, nous avons exploré des approches visant à redéfinir le prototype de la classe *NULLE*. Ces méthodes se déclinent en deux variantes : la première, **StableProto**, implique l'apprentissage direct d'une représentation vectorielle du prototype au lieu de la calculer à partir d'exemples du support set tandis que la seconde, **MultiProto**, consiste à construire plusieurs prototypes pour cette classe de manière non supervisée. Tandis que la première variante s'est révélée inefficace, la seconde a montré une légère amélioration des performances par rapport à l'approche standard qui construit un unique prototype pour la classe *NULLE*.

D'autre part, nous avons examiné des méthodes de seuillage introduisant un seuil dynamique pour déterminer si un mot doit être considéré comme un déclencheur ou non. Cette approche évite de construire un prototype pour la classe *NULLE* et propose une solution de seuillage dynamique pour cette classification. Fondamentalement, cette approche revient à considérer la classe *NULLE* comme une classe « hors domaine » ou à traiter le problème comme une détection d'anomalie. À notre connaissance, il s'agit du premier effort de recherche qui présente la détection d'événements comme une tâche d'annotation de séquences tout en traitant la classe nulle comme un problème de détection d'anomalie ou d'exemples hors domaine. L'une des difficultés supplémentaires dans notre contexte est que ces « anomalies » sont représentées en très grande majorité par rapport aux vrais exemples de déclencheurs, ce qui n'est pas la norme dans les problèmes de détection d'anomalies en général.

Les résultats expérimentaux démontrent que cette nouvelle formulation offre une amélioration significative des performances par rapport aux autres méthodes de l'état de l'art. Notamment, les méthodes de seuillage se sont révélées nettement supérieures aux autres approches que nous avons examinées. De plus, nous avons constaté que la meilleure stratégie pour déterminer un seuil dynamique est celle utilisant la fonction de répartition sur les valeurs de similarité au sein d'un épisode. Toutefois, des analyses complémentaires suggèrent que les performances pourraient encore être optimisées en trouvant un seuil encore plus adapté.

En résumé, ce chapitre a jeté les bases d'une nouvelle approche pour aborder le défi de la détection d'événements à partir de peu d'exemples en accordant une attention particulière à la gestion de la classe *NULLE*. Les résultats prometteurs que nous avons obtenus ouvrent la voie à de futures recherches dans le domaine de la détection d'événements, notamment sur le mécanisme d'apprentissage, l'utilisation d'autres types de fonction de coût ou encore d'autres stratégies de seuillage. Cette approche peut également être appliquée dans des tâches similaires telles que l'extraction d'arguments d'événements, que nous abordons dans le chapitre suivant, et d'autres tâches d'extraction d'information ayant une classe *NULLE* telle que la reconnaissance d'entités nommées ou l'extraction de relations.

Chapitre 5

Extraction des arguments d'événements par méta-apprentissage

Sommaire

Dans ce chapitre, nous abordons l'extraction des arguments d'événements à partir de peu d'exemples. La tâche d'extraction des arguments est en réalité la tâche d'intérêt de l'extraction d'événements, la détection des mentions d'événements ne servant que d'ancrage pour faciliter cette dernière.

Nous proposons une approche utilisant les réseaux prototypiques pour l'extraction d'arguments dans un contexte de faibles ressources, en modélisant la tâche comme un

problème de classification des relations entre les entités candidates et le déclencheur d'événement. De plus, nous proposons plusieurs extensions de ce modèle pour intégrer des informations syntaxiques dans la représentation des relations qui permettent d'améliorer les performances initiales.

5.1 . L'extraction des arguments d'événements à partir de peu d'exemples

Comme discuté dans le chapitre 2, le paysage général de l'extraction des arguments est assez varié : d'une part parce que la tâche d'extraction des arguments à partir de peu d'exemples a été peu abordée dans la littérature ; d'autre part parce que les approches proposées ne sont pas toujours comparables entre elles à cause de différences dans leurs formulations.

Les études se sont surtout concentrées sur l'extraction supervisée et ont montré que leurs modèles fonctionnent également dans des scénarios avec peu de données annotées. Cela est notamment dû à la capacité de généralisation de ces modèles de langue pré-entraînés. Comme nous pouvons le voir dans le tableau 5.1, la plupart de ces méthodes se fondent sur des approches génératives ou de compréhension automatique de textes (MRC). Par ailleurs, la tâche a également été abordée dans un contexte d'apprentissage sans exemple annoté. Nous comparons ces méthodes en fonction de la formulation qu'elles adoptent. ZS (pour *Zero-Shot*) correspond à la configuration d'apprentissage sans exemple, CT (pour *Class Transfert*) désigne les méthodes qui considèrent le problème d'apprentissage à partir de peu d'exemples comme un problème de transfert en classes et LR (pour *Low Ressource*) considèrent le problème comme un problème de généralisation à partir de peu d'exemples. Comme illustré dans la figure 5.1, ces méthodes reposent généralement sur des modèles pré-entraînés qui sont ensuite affinés pour l'extraction d'événements.

Typiquement, les méthodes prototypiques présentées précédemment sont des méthodes par transfert de classes dans lesquelles on entraîne les modèles sur certaines classes et on les évalue sur de nouvelles classes non vues pendant l'entraînement. Dans l'état de l'art, nous constatons que les travaux sur l'extraction des arguments se sont surtout concentrés sur des approches ZS et LR au détriment de ces méthodes par transfert de classes alors que l'on fait le constat inverse dans le cadre de la détection d'événements.

L'avantage des approches LR est qu'elles peuvent directement apprendre avec le peu d'exemples disponibles. Mais de façon générale, les modèles de langue pré-entraînés sont trop gros (en termes de nombre de paramètres) pour être entraînés avec seulement quelques exemples. Il est donc parfois préférable de travailler en ZS. Ces méthodes traduisent souvent la tâche cible en une tâche plus simple à résoudre pour le modèle de langue telle que la génération par invite (qui ressemble aux tâches de pré-

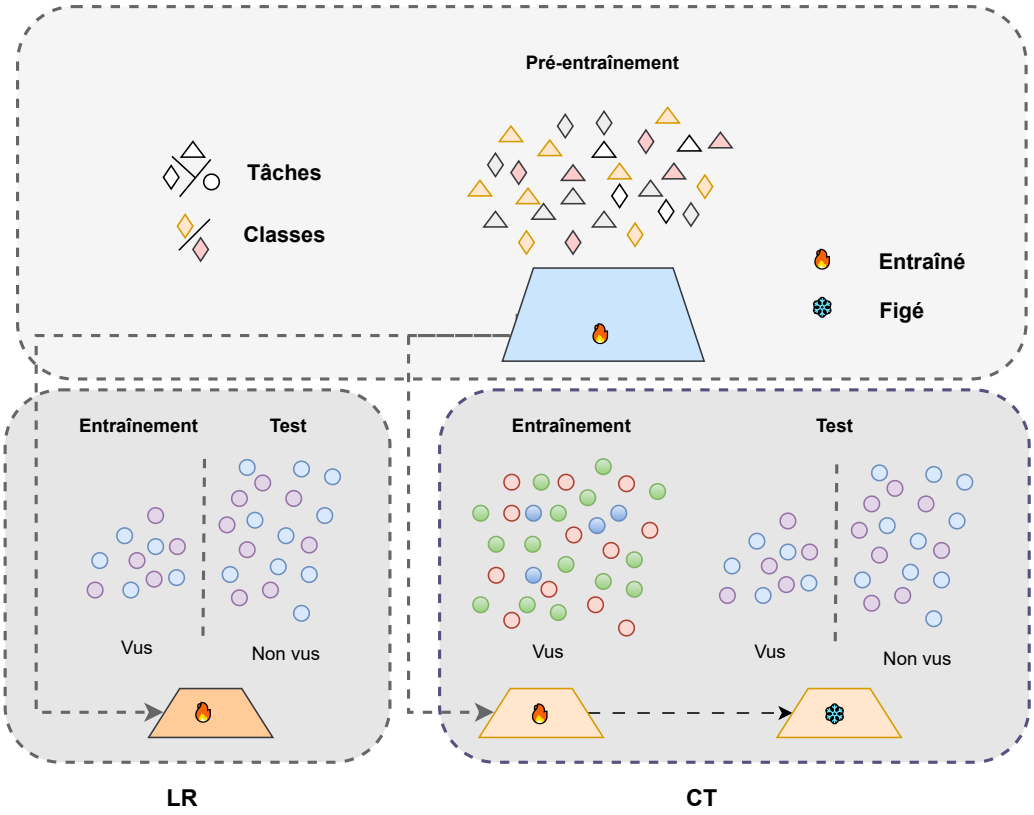


Figure 5.1 – Comparaison des formulations de la tâche d'extraction d'événements à partir de peu d'exemples.

entraînement), les approches par question/réponse ou encore l'implication de texte. Ces modèles exploitent ainsi les connaissances implicites encodées dans les modèles pré-entraînés.

Les approches CT cherchent également à faire des prédictions à partir de la connaissance de peu d'exemples, mais elles considèrent l'existence d'un jeu de données suffisamment riche sur la même tâche pour affiner le modèle.

Dans le tableau 5.1, nous répertorions les principaux travaux traitant de l'extraction des arguments à partir de peu d'exemples. Ces méthodes présentent des différences significatives dans leur formulation et leurs hypothèses concernant les ensembles de données et leurs méthodes d'évaluation.

5.2 . Formulation du problème

Comme pour la détection d'événements, que nous avons traitée dans les deux chapitres précédents, nous proposons d'aborder l'extraction des arguments à partir de peu d'exemples à l'aide de réseaux prototypiques. Ce choix est principalement motivé par l'efficacité démontrée de ces méthodes dans la tâche de détection d'événements et dans plusieurs autres travaux en extraction d'information (Gao et al., 2019a; Fritzler et al., 2018; Lai et al., 2021a). Notre objectif ici est d'évaluer les capacités de ces approches prototy-

Modèle	Méthode	Res. Ext.	Hypothèses		Config.		
			Trig.	Ent./Arg.	ZS	LR	CT
Lin et al. (2023)	GEN		✓	✓	✓		✓
Ma et al. (2022)	GEN		✓			✓	
Hsu et al. (2022)	GEN					✓	
Zhang et al. (2022f)	CLF	✓	✓		✓		✓
Sainz et al. (2022a)	MRC	✓	✓		✓	✓	
Dai et al. (2022)	GEN			✓		✓	
Yu et al. (2022)	CLF	✓			✓		✓
Zhang et al. (2021a)	CLF		✓		✓	✓	✓
Zhou et al. (2021)	MRC		✓			✓	
Lyu et al. (2021)	MRC				✓		✓
Chen et al. (2020)	MRC				✓	✓	
Huang et al. (2018)	CLF	✓			✓		✓

Table 5.1 – Comparaison des différentes configurations pour l'extraction d'événements à partir de peu d'exemples. CLF, GEN et MRC sont respectivement les méthodes par classification, par génération ou par compréhension du langage. **Res. Ext.** correspond à l'utilisation de ressources externes. **Trig.** et **Ent./Arg.** désigne le fait de prendre en entrée les déclencheurs, les entités ou les arguments annotés.

piques dans le cadre de l'extraction des arguments étant donné que cela a été très peu exploré dans la littérature. Un second objectif qui en découle naturellement est de proposer un cadre d'évaluation N-ways, k-shots (Vinyals et al., 2016) pour la tâche d'extraction d'événements.

L'extraction des arguments d'événement peut également être traitée comme une tâche d'annotation de séquence au format BIO où, au lieu d'annoter les déclencheurs, nous annotons les arguments. Nous donnons un exemple d'une telle annotation dans le tableau 5.2. Cela peut également être vu comme une tâche de classification de mots, exactement comme pour la tâche de détection d'événement. La seule différence ici est que cette tâche est bien plus difficile, car, en général, les occurrences d'événement contiennent plus d'arguments que de déclencheurs et ces arguments sont bien plus complexes (souvent en plusieurs mots). Nous proposons de formuler comme une tâche de classification de relations entre le déclencheur et les entités en supposant qu'un rôle d'argument est une relation particulière entre le déclencheur et l'argument.

The	police	officer	who	fired	into	a	car	full	of	teenagers	was	fired	Tuesday	.
B-Person	I-Person	I-Person	O	O	O	O	O	O	O	O	B-End-Pos.	I-End-Pos.	B-Time	O

Table 5.2 – Exemple d'annotation BIO pour la phrase : *The police officer who fired into a car full of teenagers was fired Tuesday.* vis-à-vis de l'événement de licenciement (*End-Position*). L'argument *Person* désigne la personne licenciée et *Time*, un marqueur temporel du licenciement.

5.2.1 . Apprentissage *N-ways, k-shots* pour l'extraction des arguments

Nous proposons d'aborder la tâche par apprentissage épisodique N-ways, k-shots comme pour la détection d'événements, mais avec une légère variation. En effet, bien que la classification soit effectuée sur les arguments des événements, nous considérons les nouvelles classes au niveau des types d'événement. Cela correspond au scénario dans lequel les types d'événements de l'ensemble d'évaluation n'ont pas été rencontrés lors de l'entraînement, ce qui est davantage en phase avec les applications du monde réel où de nouveaux événements peuvent apparaître plutôt que de nouveaux rôles d'argument pour des événements existants. Par conséquent, notre formulation de l'approche N-ways, k-shots inclut un N variable représentant le nombre d'arguments pour un type d'événement donné, chacune ayant k instances dans le support set. Par ailleurs, plusieurs types d'événement peuvent contenir les mêmes types de rôle d'argument et donc, certains rôles peuvent déjà avoir été vus lors de l'entraînement du modèle. Par exemple, dans la figure 5.2, on peut voir que l'argument *Agent* est commun à 8 types d'événements.

Néanmoins, nous pouvons également considérer qu'un argument pris dans le contexte d'un type d'événement particulier est différent du même argument dans le contexte d'un autre type. Par exemple, l'argument « *Agent* » peut être interprété différemment lorsqu'il s'agit du type d'événement « Die » (c'est-à-dire l'agent attaquant) par rapport à l'événement de type « Transport » (qui désigne l'agent responsable du transport). Par ailleurs, certains autres arguments peuvent être proches dans leurs sémantiques, bien qu'ils n'aient pas le même nom. Par exemple, « Attaker » est l'agent attaquant dans un événement « Attack », ce qui est sémantiquement proches des arguments « Agent » dans les événements « Die » ou « Injure ». Nous sommes en réalité dépendants des définitions adoptées dans les guides d'annotation des jeux de données.

Dans ce contexte, chaque épisode se présente comme une tâche de classification, notée $\mathcal{T} = \{\mathcal{S}, \mathcal{Q}\}$. Chaque tâche comporte un support set, $\mathcal{S}$, et un query set, $\mathcal{Q}$. Le support set $\mathcal{S}$ comprend un type d'événement avec ses N classes d'arguments, chacun étant représenté par k instances annotées tandis que le query set $\mathcal{Q}$ contient une phrase mentionnant au moins un événement du même type.

L'objectif d'un épisode est de réaliser la détection des arguments de l'événement dans la phrase du query set en s'appuyant sur les informations fournies par le support set. Pour garantir que l'ensemble de support contienne exactement k exemples, nous effectuons une sélection de manière à ce qu'il y ait au moins un exemple d'argument pour le type d'événement considéré. Chaque phrase peut contenir plusieurs entités ayant des rôles différents ou non, voire n'ayant pas de rôle du tout. Lorsque le nombre d'entités pour un rôle donné dépasse k, nous en conservons uniquement k et ignorons les autres. Nous ne prenons pas en compte les arguments ne pouvant pas être couverts par au moins k exemples. Cet échantillonnage est réalisé de manière à obtenir le nombre minimal de phrases distinctes dans le support set.

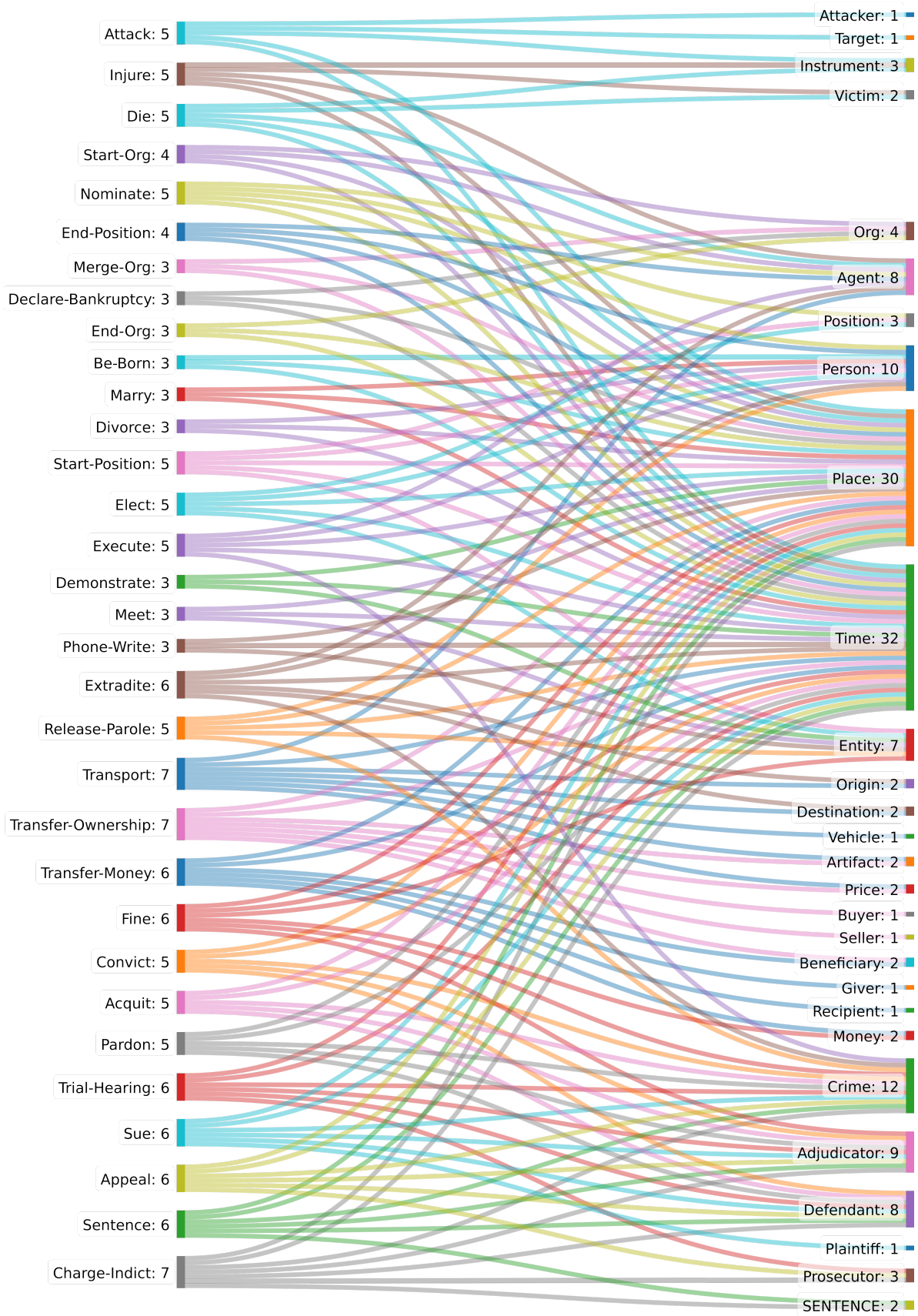


Figure 5.2 – Les types d'événements (à gauche) et leurs arguments (à droite) du jeu de données ACE-2005. Le chiffre devant chaque type d'événement indique le nombre de ses arguments et le nombre devant chaque argument indique le nombre d'événements dont il est argument.

5.2.2 . Traitement des instances

Pour un type d'événement e donné, une instance est définie par (x_i, y_i) avec $x_i = (s_i^e, tr_i^e, a_i)$, où s_i^e est la phrase mentionnant l'événement, tr_i^e le déclencheur, a_i le candidat argument, et y_i le rôle appartenant à $\mathcal{A}^e = \{\mathcal{A}_+^e \bigcup NULLE\}$, avec $\mathcal{A}_+^e$ étant l'ensemble des arguments du type d'événement e et $NULLE$ indiquant que l'entité n'a aucun rôle dans l'événement. La même phrase peut donc appartenir à autant d'exemples qu'elle contient de mention d'entités. Nous prenons en compte toutes les mentions d'entités même si, parfois, il peut y avoir une coréférence entre deux mentions qui désignent la même entité. Comme pour la détection d'événements, nous avons une classe pour les entités n'ayant pas de rôle appelée classe *NULLE*.

Cette approche suppose donc d'avoir des entités candidates préalablement identifiées afin de leur attribuer un rôle d'argument ou les classer comme appartenant à la classe *NULLE*. Dans cette étude, nous adoptons une approche similaire à celle de certains travaux antérieurs (Yu et al., 2022; Lin et al., 2023; Sainz et al., 2022a; Dai et al., 2022), en considérant que les entités annotées dans le jeu de données ACE-2005 sont préalablement connues. Cette modélisation, bien que reposant sur la présupposition de la connaissance des entités nommées, peut être complétée par un système d'extraction d'entités candidates (NER) en amont, ou un modèle d'annotation en rôles sémantiques (SRL) pour effectuer l'extraction de bout en bout.

Suivant plusieurs travaux en extraction d'information (Zhang et al., 2019c; Han et al., 2018; Baldini Soares et al., 2019; Gao et al., 2019a), nous utilisons également des jetons (*tokens*) spéciaux pour marquer les entités et le déclencheur, ce qui permet de porter plus d'attention sur ces éléments importants. Pour obtenir les représentations vectorielles du déclencheur et de l'entité, nous prenons la moyenne des représentations des mots qui les composent.

En définitive, la phrase présentée dans l'exemple 5.2 se décline en deux exemples différents correspondant chacun à l'une des entités qu'elle contient (voir tableau 5.3).

Exemple																		Label
<E>	The	police	officer	</E>	who	fired	into	a	car	full	of	teenagers	<T>	was	fired	</T>	Tuesday	.	Person
The	police	officer	who	fired	into	a	car	full	of	teenagers	<T>	was	fired	</T>	<E>	Tuesday	</E>	.	Time

Table 5.3 – Deux exemples construits à partir de la phrase *The police officer who fired into a car full of teenagers was fired Tuesday.* vis-à-vis de l'événement « *End-Position* ».

5.3 . Modélisation de l'extraction d'arguments comme un problème de classification de relations entre le déclencheur et les entités

Inspiré par des travaux sur l'extraction de relations à partir de peu d'exemples (Gao et al., 2019a; Han et al., 2018), nous abordons l'extraction des arguments en la conceptua-

lisant comme une tâche de classification de relations. Cette perspective nous permet de considérer les rôles comme une catégorie spécifique de relation qui lie le déclencheur événementiel aux arguments.

Notre modèle prend en entrée une instance $x_i = (s_i^e, tr_i^e, a_i)$, composée d'une mention d'événement, du déclencheur de l'événement et de l'entité candidate à laquelle nous souhaitons attribuer un rôle. Cette instance est traitée par un encodeur pour produire une représentation vectorielle $h_i = \mathcal{E}(x_i)$ pour chaque paire déclencheur-entité dans le contexte d'un événement donné. En pratique, cette représentation est obtenue en concaténant les représentations contextuelles du déclencheur et de l'entité. Ensuite, cette représentation est classifiée en relation à l'aide d'un algorithme de méta-apprentissage. Nous utilisons en l'occurrence un réseau prototypique standard et la variante contrastive que nous avons présentée dans le chapitre précédent.

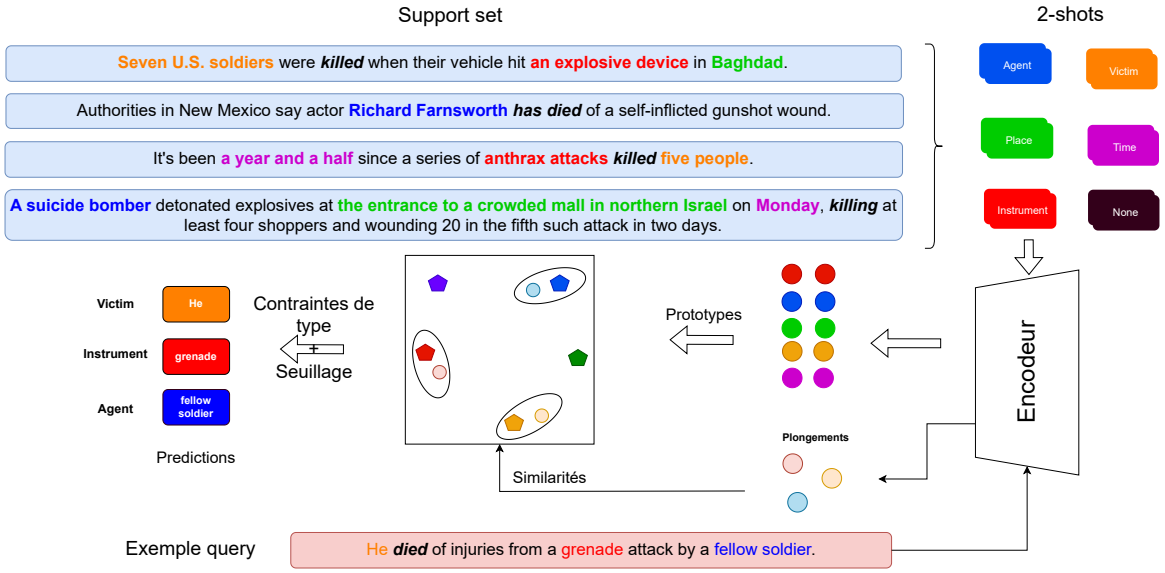


Figure 5.3 – Vue d'ensemble de notre modèle. Les déclencheurs sont en ***gras italique*** et chaque couleur correspond à une classe d'argument.

Nous donnons un aperçu général de notre approche dans la figure 5.3.

Comme souligné précédemment dans le Chapitre 3, le choix de l'encodeur revêt d'une importance capitale au sein de ces réseaux prototypiques. En effet, la capacité de cet encodeur à produire des représentations vectorielles de haute qualité pour chaque classe (c'est-à-dire le prototype), influence directement la performance globale du système. Il est essentiel que ces représentations soient suffisamment discriminantes pour permettre au modèle de différencier efficacement les différentes classes.

5.3.1 . Enrichissement des représentations des relations

Nous proposons d'explorer l'exploitation des relations syntaxiques entre le déclencheur et une entité pour aider à distinguer les entités qui présenteraient des représen-

tations contextuelles similaires. En effet, nous avons observé que certaines entités pouvaient être confondues au sein d'un même événement, notamment lorsque leurs rôles se ressemblent ou sont symétriques. Par exemple, dans un contexte d'attaque, l'entité représentant l'agent attaquant peut parfois être confondue avec celle représentant la cible de l'attaque. De même, dans un contexte de transport, les entités représentant l'origine et la destination peuvent être sujettes à confusion. Notre approche vise donc à utiliser des informations syntaxiques additionnelles pour lever ces ambiguïtés. Nous croyons que les informations syntaxiques peuvent jouer un rôle clé dans cette désambiguïsation, car les rôles des entités au sein d'un événement sont souvent étroitement liés à leurs rôles syntaxiques dans les phrases qui les décrivent. Dans l'exemple illustré de la figure 5.4, le déclencheur « fired » est lié à son argument « police officer » par une relation syntaxique de sujet (en voix passive) alors qu'ils sont distants dans la phrase.

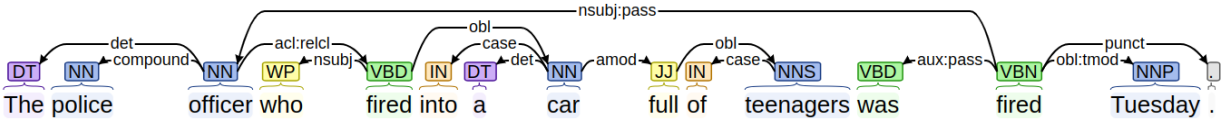


Figure 5.4 – Analyse en dépendances syntaxiques et étiquettes de parties du discours pour la phrase : *The police officer who fired into a car full of teenagers was fired Tuesday.*. Cette analyse a été produite avec CoreNLP [1] et la visualisation est générée avec l'outil d'annotation Brat [2]

Ces informations syntaxiques ont par ailleurs déjà été utilisées avec succès dans des études antérieures pour l'extraction d'événements, démontrant leur capacité à améliorer de manière significative les performances des modèles (Nguyen and Grishman, 2018a; Balali et al., 2020). Dans le travail de Nguyen and Grishman (2018a), ces informations sont exploitées dans le cadre de la détection d'événements en utilisant des CNN. Plus précisément, elles sont utilisées dans le mécanisme d'agrégation des vecteurs de convolution, contrairement aux approches classiques qui utilisaient un *Max-Pooling* (Nguyen and Grishman, 2015; Chen et al., 2015b). De même, dans l'étude menée par Balali et al. (2020), l'exploitation du plus court chemin syntaxique entre les arguments dans le graphe de dépendances syntaxiques a permis d'améliorer à la fois la détection des déclencheurs et l'extraction des arguments.

L'une des contributions de ce chapitre consiste à examiner des techniques visant à enrichir ces représentations à l'aide d'informations syntaxiques sur les phrases. Plus spécifiquement, nous avons exploré deux approches d'enrichissement : une approche statique par concaténation (**BERT++**) et une approche dynamique impliquant l'utilisation de réseaux de convolution sur des structures de graphes (**BERT-GCN**).

Nous utilisons l'encodeur BERT (Devlin et al., 2019b) comme point de comparaison de référence pour évaluer les avantages de l'injection d'informations syntaxiques.

BERT++ : intègre de manière statique les informations syntaxiques en associant un vecteur à chaque type de relation syntaxique, ces vecteurs étant ajustés pendant l'apprentissage du modèle. Plus précisément, cette approche consiste à concaténer les représentations vectorielles des étiquettes morphosyntaxiques (PoS-tags) de l'entité ainsi que du chemin de dépendance syntaxique entre le déclencheur et l'entité aux plongements contextuels du déclencheur et de l'entité. Compte tenu de la variabilité de la longueur du chemin de dépendance entre le déclencheur et l'entité, nous avons appliqué une agrégation *Max-pooling* sur l'ensemble des étiquettes de dépendance syntaxique afin d'obtenir les plongements des chemins de dépendance syntaxique.

Cette intégration d'informations syntaxiques aux plongements contextuels de BERT permet au modèle d'acquérir une meilleure compréhension des rôles potentiels des entités, réduisant ainsi les confusions lors de l'extraction des arguments. Nous avions également envisagé d'ajouter un vecteur correspondant au type des entités mais cela a eu un impact négatif sur les performances globales du modèle.

BERT-GCN : combine les informations syntaxiques avec les plongements fournis par BERT en utilisant un réseau de convolution sur les graphes (GCN) (Kipf and Welling, 2017b). Cette méthode nous permet de bénéficier des avantages des représentations contextuelles fournies par BERT ainsi que des informations structurées présentes dans les dépendances syntaxiques. Nous avons opté pour une variante des GCN appelée RGCN (*Relational Graph Convolutional Network*) (Schlichtkrull et al., 2018; Xu and Yang, 2019), qui fait une distinction entre les différents types de relations présentes dans le graphe et apprend un filtre de convolution spécifique pour chaque type de relation. Cette approche est particulièrement pertinente dans notre cas car elle permet de différencier les relations syntaxiques, contribuant ainsi à réduire les confusions entre des arguments similaires. Cependant, cette modélisation entraîne une augmentation significative du nombre de paramètres à apprendre, correspondant aux paramètres du GCN.

De manière formelle, le plongement de chaque mot i est obtenu en utilisant son plongement initial fourni par le modèle BERT, qui est ensuite soumis à L couches du RGCN. La sortie de la couche l est donnée par :

$$h_i^{(l)} = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_i^r}\frac{1}{c_{i,r}}W_r^{(l)}h_j^{(l-1)} + W_0^{(l)}h_i^{(l-1)}\right) \tag{5.1}$$

où h_i^0 est le plongement contextuel fourni par BERT, $\mathcal{N}_i^r$, l'ensemble des nœuds voisins du nœud i ayant la relation $r \in \mathcal{R}$, $c_{i,r} = |\mathcal{N}_i^r|$ et $\sigma()$, une fonction d'activation sigmoïde.

Afin de permettre à l'encodeur BERT de capturer des informations sur l'emplacement du déclencheur et de l'entité au sein d'une phrase, nous marquons le début et la fin de ces éléments avec des jetons spéciaux, comme suggéré dans d'autres travaux de la

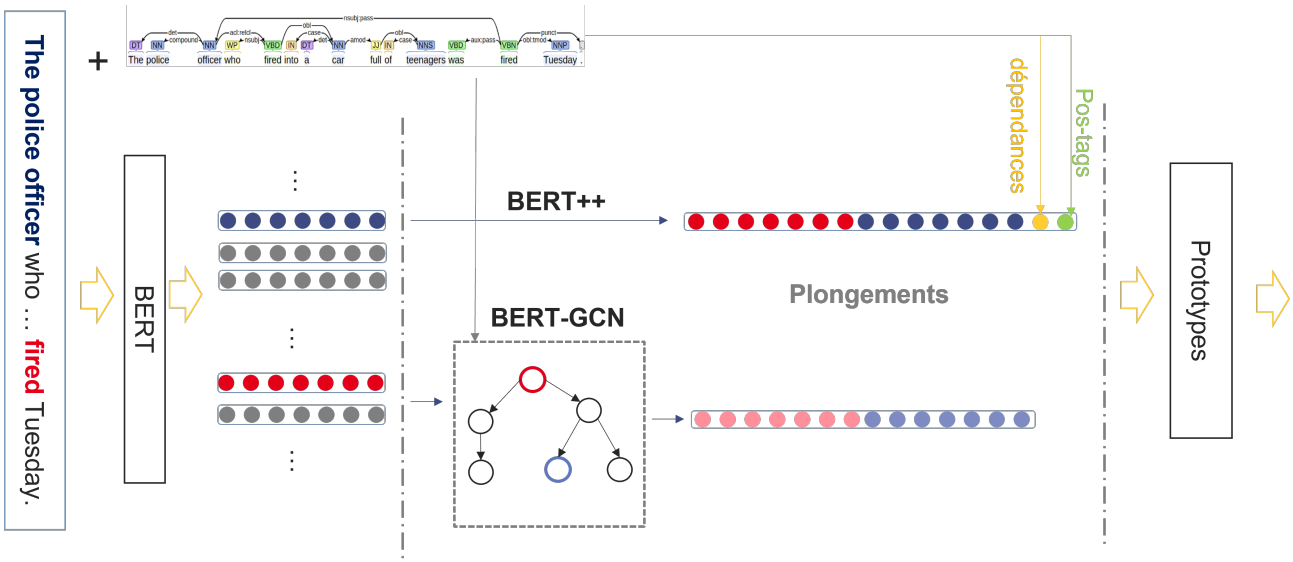


Figure 5.5 – Procédure d'encodage des paires déclencheur/entité.

littérature (Zhang et al., 2019c; Han et al., 2018; Baldini Soares et al., 2019). Ensuite, nous passons l'ensemble du passage à l'encodeur pour obtenir un plongement de chaque mot. Nous concaténons enfin les plongements du déclencheur et de l'entité pour obtenir un plongement du rôle considéré. Ce processus de codage est synthétisé à la figure 5.5.

5.3.2 . Module de classification

Le module de classification vise à classifier les instances en fonction de leurs similarités avec les représentations des prototypes de chaque classe. À cette fin, nous avons mené des expériences avec deux approches : les Réseaux Prototypiques standard (*Prototypical Networks*, **Proto**) et leur version contrastive (*Contrastive Prototypical Networks*, **C-Proto**), présentée dans le chapitre précédent.

La différence entre ces deux modèles réside dans le fait que **C-Proto** adopte un apprentissage contrastif et ne construit pas de prototype pour la classe *NULLE*, qui regroupe les entités n'ayant pas de rôle dans les événements. Dans ce cas, les exemples de cette classe sont filtrés à l'aide d'un seuil de similarité. Pour rappel, le modèle **Proto** est entraîné à l'aide d'une fonction de coût de type entropie croisée (*cross-entropy loss*), combinée avec une fonction *softmax* (voir Équation 5.2).

$$P(y_i = y|x_i, \mathcal{S}) = \frac{\exp(s(h_i, c^y))}{\sum_{j \neq y} \exp(s(h_i, c^j))} \quad \text{et} \quad \mathcal{L}_{CE} = -\sum_{(x_i, y_i) \in \mathcal{Q}} \log\left(P(y_i|x_i, \mathcal{S})\right) \tag{5.2}$$

Le modèle **C-Proto** est, quant à lui, entraîné avec une fonction de coût de charnière (*hinge loss*) introduite dans le chapitre 4 (voir équations 5.3 et 5.4). Cette fonction de coût est la somme de deux termes : l'un ($\mathcal{L}_+$) concerne uniquement les arguments et l'autre

($\mathcal{L}_-$), les entités non-arguments (nous notons $\mathcal{Q}^+$ le sous-ensemble des arguments du query set d'un épisode et $\mathcal{Q}^-$, le sous-ensemble du query set contenant des entités non-arguments).

$$\mathcal{L}_+(\mathcal{S}, \mathcal{Q}) = \sum_{(x_i, y_i) \in \mathcal{Q}^+} \sum_{j \neq y_i} \max_j(0, \mathcal{M}_0 - s(h_i, c^j) + s(h_i, c^{y_i})) \tag{5.3}$$

$$\mathcal{L}_-(\mathcal{S}, \mathcal{Q}) = \max_{(x_i, y_i) \in \mathcal{Q}^+}(0, \max_{x_j \in \mathcal{Q}^-}(s\,(h_j, c^{y_i}) - \mathcal{M}_1)) \tag{5.4}$$

Contrairement à la proposition du chapitre 4, qui utilise une fonction de répartition pour estimer le seuil, nous calculons le seuil ici en utilisant la valeur de similarité trouvée sur l'exemple le plus proche dans l'ensemble de support. Cette variation est due au fait que, dans le cas de l'extraction des arguments, certaines phrases peuvent ne contenir qu'une seule entité candidate ou peuvent ne contenir que des entités correspondant à des non-arguments [3]. Par conséquent, l'utilisation de la fonction de répartition dans le cas d'une seule entité ou d'entités qui n'ont pas de rôle ne permettrait pas fixer un seuil raisonnable. Nous voudrions, dans certains cas, que ce seuil soit supérieur aux similarités pour toutes les entités. Ce problème ne se posait pas dans le cas des déclencheurs parce que nous étions sûrs d'avoir au moins un déclencheur par phrase.

5.3.3 . Contraintes de compatibilité entre les types des entités et leurs rôles

Suivant des travaux antérieurs sur l'extraction d'événements (Sainz et al., 2022a; Lin et al., 2023), nous utilisons la connaissance préalable des types des entités pour contraindre les prédictions des rôles d'arguments. Pour intégrer cette connaissance du domaine, disponible dans le guide d'annotation, nous associons chaque rôle d'argument à la classe la plus proche ayant un type compatible. Par exemple, si une entité est de type « Personne », alors les rôles « Agent attaquant » ou « Victime » sont considérés comme compatibles, tandis que le rôle « Lieu » ne l'est pas. Ce traitement supplémentaire nous permet d'incorporer une contrainte sémantique dans le processus d'extraction et contribue à améliorer la cohérence et la précision des prédictions en s'assurant que les rôles attribués aux entités sont cohérents avec leurs types. Cela nous permet également de mieux gérer les situations dans lesquelles plusieurs rôles d'argument pourraient sembler appropriés pour une entité donnée, en sélectionnant seulement ceux qui sont compatibles. Les contraintes de compatibilité entre les types d'entité et les rôles d'argument sont illustrées à la figure 5.6.

En pratique, lors de l'évaluation du modèle, nous commençons par vérifier la compatibilité entre le rôle prédit et le type de l'entité candidate. Si la contrainte n'est pas

3. Dans ces cas, les arguments de la mention considérée se trouvent souvent dans d'autres phrases. Il faudrait une extraction au niveau du document pour les identifier.

satisfaite, nous considérons la prédiction comme incorrecte et prenons la classe du prototype suivant le plus proche jusqu'à ce que le rôle prédit corresponde au type d'entité ou à la classe *NULLE*, qui n'est soumise à aucune contrainte.

5.4 . Expérimentations

Jeu de données

Nous avons mené nos expériences sur l'ensemble de données ACE-2005 (Walker et al., 2006) [4], avec la partition fournie par Lai et al. (2021b). Cette partition garantit qu'il n'y a pas de chevauchement entre les types d'événements dans les ensembles d'entraînement et d'évaluation, simulant ainsi un scénario réaliste avec une faible disponibilité des données (voir l'annexe A pour plus de détails).

Paramètres expérimentaux

Nous utilisons l'encodeur `BERT-large-uncased` pour fournir les représentations des mots à partir des phrases en entrée. De plus, pour l'encodeur **BERT++**, nous utilisons des vecteurs entraînables de taille 256 pour encoder les dépendances syntaxiques et les étiquettes de partie du discours (PoS) obtenues à l'aide de SpaCy [5]. Nous optons pour seulement deux couches de convolution au niveau du RGCN car c'est ce qui donne le meilleur résultat empirique (voir figure 5.7). Ce nombre limité de couches peut être expliqué de la façon suivante : à mesure que le nombre de couches augmente, le modèle prend en compte un voisinage de plus en plus vaste. Étant donné que le graphe de dépendances syntaxiques est entièrement connecté, au-delà d'un certain nombre de couches, tous les nœuds peuvent être considérés comme appartenant au même voisinage. Dans cette configuration, le plongement de chaque nœud se rapproche de la moyenne des plongements de ses voisins, ce qui peut conduire à une perte d'informations discriminantes.

5.4.1 . Résultats

Nous reportons nos principaux résultats dans le tableau 5.4. Les entités considérées pour l'extraction des arguments sont les entités annotées dans le jeu de données ACE-2005.

Ces résultats nous permettent de tirer deux conclusions principales. D'une part, quel que soit l'encodeur considéré, nous pouvons observer que la version contrastive **C-Proto** affiche des performances légèrement supérieures à celles du réseau prototypique standard. Cette observation vient confirmer les constatations faites dans le chapitre précédent (chapitre 4) sur l'efficacité de cette approche, en particulier en ce qui concerne la gestion de la classe *NULLE*. Toutefois, les apports dans ce cadre sont bien moindres que

4. Les autres jeux de données utilisés dans le chapitre 4 n'ont pas d'annotation sur les arguments.
5. SpaCy

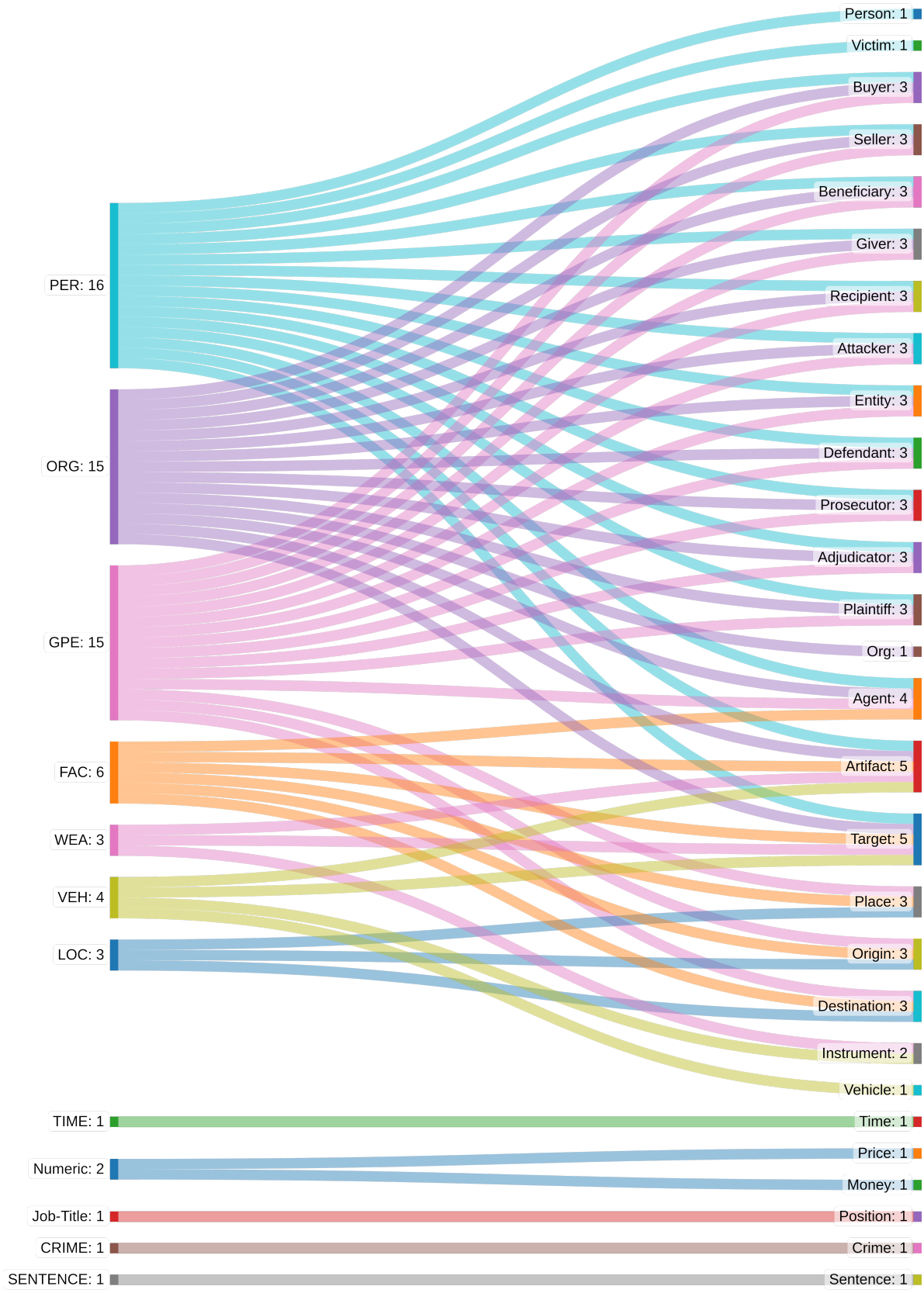


Figure 5.6 – Correspondance entre les types des entités et leurs rôles. Le nombre devant chaque entrée (entité ou rôle) correspond au nombre d'entrées auxquelles elle est connectée.

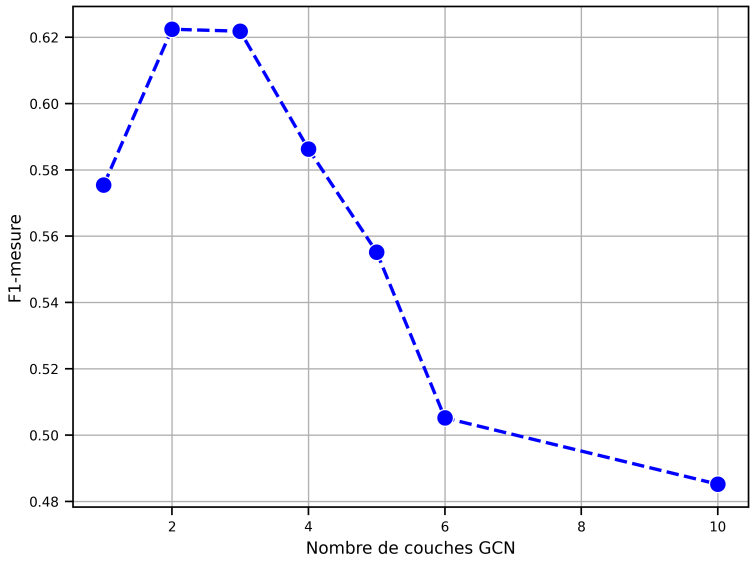


Figure 5.7 – F1-mesure en fonction du nombre de couches de convolution dans le GCN.

		5-shots			10-shots		
Encodeur	Modèle	P	R	F1	P	R	F1
BERT	Proto	63,1 ± 0,9	56,4 ± 1,0	59,6 ± 0,5	66,4 ± 0,5	61,6 ± 0,7	63,9 ± 0,3
	C-Proto	62,7 ± 0,9	57,0 ± 1,2	60,0 ± 1,0	67,1 ± 0,8	63,8 ± 0,9	65,5 ± 0,8
BERT++	Proto	64,9 ± 1,1	58,6 ± 1,2	61,6 ± 0,8	66,8 ± 1,5	63,8 ± 1,1	65,2 ± 0,6
	C-Proto	65,8 ± 0,5	58,8 ± 1,8	62,1 ± 1,0	66,8 ± 1,7	**66,5*** ± **1,7**	**66,7*** ± **1,0**
RGCN	Proto	**69,0** ± **2,1**	56,6 ± 4,0	62,2 ± 2,2	**71,2*** ± **0,7**	60,0 ± 1,5	65,0 ± 0,9
	C-Proto	68,5 ± 1,1	**59,2*** ± **1,7**	**63,5*** ± **1,2**	69,2 ± 0,5	61,4 ± 0,8	65,1 ± 0,5

Table 5.4 – Résultats de l'extraction des arguments d'événements : Précision (P), Rappel (R) et F1-mesure (F1). Nos meilleurs scores sont en **gras** et les seconds meilleurs sont soulignés. * indique que le meilleur score est statistiquement significatif par rapport au deuxième.

pour la tâche de détection d'événements. Cela peut être lié à la difficulté relative de la tâche d'extraction des arguments par rapport à l'extraction des déclencheurs. En effet, les entités occupant un même rôle n'ont pas forcément de similarités sémantiques entre elles, contrairement aux déclencheurs, qui appartiennent souvent au même champ lexical.

D'autre part, nous voyons que l'intégration des informations syntaxiques améliore les performances dans tous les cas. Cette constatation suggère que l'exploitation d'informations syntaxiques pour enrichir la relation entre les déclencheurs d'événements et leurs arguments est bien bénéfique pour la tâche d'extraction des arguments d'événements. Par ailleurs, l'intégration dynamique **BERT-GCN** semble plus efficace que l'intégration statique **BERT++** lorsque très peu de données sont disponibles (5-shots).

5.4.2 . Comparaison avec l'état de l'art

Comme discuté dans la revue de la littérature, les méthodes de l'état sont assez peu comparables entre elles. Les méthodes qui présentent des résultats pour l'extraction

d'arguments à partir de peu d'exemples adoptent généralement une approche consistant à n'utiliser qu'un pourcentage de l'ensemble d'entraînement pour simuler un scénario d'apprentissage avec peu d'exemples. Même si ces pourcentages sont connus, on ne sait pas précisément sur quels exemples ces méthodes ont été entraînées. Dans ce contexte, les différences dans les performances peuvent simplement venir du biais de sélection sur les données d'apprentissage et non des modèles en eux-mêmes. De plus, ces modèles ne font pas exactement les mêmes hypothèses sur les informations en entrée des modèles (connaissances du déclencheur, des entités par exemple), ni même sur le nombre de rôles à prendre en compte. Certains travaux ne considèrent ainsi pas les rôles attributs tels que les lieux et dates. En effet, le jeu de données ACE-2005 fait une distinction entre les rôles en les classant en deux catégories : les arguments désignant les participants à l'événement (les personnes, organisations, etc.) et les attributs servant à décrire l'événement (les lieux, dates, etc.).

Par ailleurs, cette formulation diffère de la nôtre, qui se concentre plutôt sur un scénario d'adaptation au domaine, où de nouvelles classes d'événements non observées lors de l'apprentissage du modèle peuvent apparaître. Nous considérons que l'approche par transfert de classes est plus réaliste dans un contexte applicatif réel car les besoins en apprentissage à partir de peu d'exemples sont précisément motivés par l'apparition de nouveaux types d'événements. Compte tenu de la modélisation spécifique que nous avons adoptée pour aborder ce problème ainsi que des hypothèses sous-jacentes, notre modèle n'est pas non plus comparable avec ces méthodes. Cependant, nous souhaitons tout de même fournir un aperçu de certains résultats issus de la littérature afin d'évaluer la pertinence de notre travail par rapport aux approches existantes.

Notre objectif principal dans ce travail n'était pas nécessairement de surpasser les performances de l'état de l'art mais plutôt de réaliser une étude exploratoire de l'extraction d'arguments en utilisant une approche par méta-apprentissage. De plus, nous souhaitions examiner l'impact des informations syntaxiques dans ce contexte. Cette étude fournit ainsi un cadre d'évaluation pour l'extraction des arguments à partir de peu d'exemples au niveau phrastique. À notre connaissance, il s'agit de la première tentative de ce genre, ouvrant ainsi de nouvelles perspectives de recherche dans ce domaine en constante évolution.

À titre indicatif, nous présentons les résultats des méthodes qui se rapprochent le plus de la nôtre en termes d'hypothèses. Cela couvre les méthodes partant de la connaissance du type d'événement et du déclencheur et qui effectuent une évaluation type par type. Nous considérons d'une part, les modèles supposant la connaissance des entités annotées et d'autre part, ceux présupposant la connaissance des arguments, ce qui implique l'absence d'une classe *NULLE*.

Méthodes considérant la donnée des entités

Nous nous comparons aux modèles **PAIE** (Ma et al., 2022), **BIP** (Dai et al., 2022) et **NLI** (Sainz et al., 2022a).

PAIE est une approche générative utilisant le modèle de langue BART (Lewis et al., 2020) pour l'extraction des arguments. L'une des particularités de cette approche est qu'elle permet de prédire des empans (*spans*) plutôt que les entités en elles-mêmes, limitant ainsi le phénomène d'hallucination. Néanmoins, la sélection du début et de la fin de chaque empan est conditionnée par la réponse générée par le modèle de langue. De plus, ce modèle utilise des amorces flexibles (*soft prompt*) permettant d'identifier plusieurs arguments pour le même rôle, ce qui n'est pas possible avec des amorces fixes. Concrètement, il aborde la tâche d'extraction comme une tâche de traduction automatique en cherchant à traduire une mention d'événement en un schéma d'événement. Dans ce contexte, un schéma d'événement est une phrase à trous où chaque emplacement correspond à un rôle. L'objectif est d'aligner les entités de la phrase sur les emplacements correspondant dans le schéma, chaque emplacement correspondant à un rôle d'argument.

BIP est un autre modèle génératif qui remplit des patrons d'événements, mais dans ce cas, de façon itérative et bidirectionnelle. Cette approche présente deux avantages significatifs : d'abord, le remplissage itératif des patrons permet d'utiliser les prédictions précédentes pour améliorer les prédictions courantes, ce qui contribue à affiner le résultat au fur et à mesure des prédictions. En quelque sorte, le modèle garde en mémoire les prédictions déjà faites et prend sa décision en fonction d'elles. Ensuite, la bidirectionnalité de l'approche permet de prendre en compte à la fois le contexte avant et après l'entité considérée, ce qui enrichit la compréhension globale de l'événement.

Enfin, l'approche **NLI** (Sainz et al., 2022a) propose une méthode innovante qui transforme la tâche d'extraction d'arguments en une tâche d'implication textuelle (*Textual Entailment*) (Dagan et al., 2006). Cette méthode vise à vérifier la cohérence entre une phrase hypothèse et une phrase prémisse. Pour un type d'événement et une entité donnés, l'approche consiste à utiliser la mention d'événement comme prémisse et à considérer comme hypothèse le fait que cette entité occupe l'un des rôles de l'événement en question. L'hypothèse est confirmée seulement si le modèle prédit une implication. Cette méthode tient compte des compatibilités entre les types d'entités et les rôles pour réduire le nombre d'hypothèses à vérifier. Ce processus est original et peut être mis en œuvre sans nécessiter un entraînement du modèle (*Zero-Shot*). De plus, cette méthode a été adaptée avec succès à plusieurs autres tâches d'extraction d'informations dans le système ZS4IE (Sainz et al., 2022b) [6].

Les résultats de ces méthodes sont illustrés par la figure 5.8. Ces résultats sont donnés en considérant un certain pourcentage des données d'entraînement lors de la formation des modèles. Bien que notre méthode ne soit pas directement comparable avec

6. ZS4IE

ces méthodes, nous estimons que notre configuration 5-shots (c'est-à-dire 5 exemples par rôle) correspond en quantité à environ 3% des données d'évaluation (voir tableau 5.5). Il faut toutefois noter que nous nous entraînons sur 18 types d'événements et tous leurs exemples, les 3% de données ne concernent que les types non vus pendant l'entraînement.

Split	#Types	#Events	#Roles	#Ents. (Args)
Train	18	4 736	19	29 392 (9 566)
Dev	11	1 980	17	10 583 (3 049)
Test	11	1 909	17	11 540 (3 248)

Table 5.5 – Notre découpage sur le jeu de données ACE-2005.

La figure 5.8 met en évidence la compétitivité de notre approche par rapport aux méthodes de l'état de l'art jusqu'à environ 5% des données d'entraînement. Ces résultats suggèrent que notre approche par méta-apprentissage est bien prometteuse dans le domaine de l'extraction d'arguments d'événements à partir de peu d'exemples.

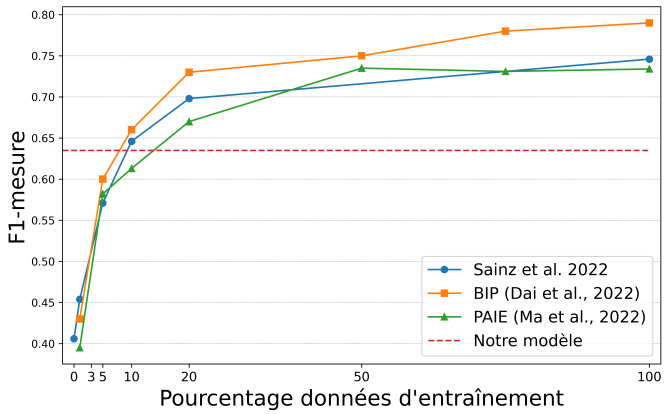


Figure 5.8 – Comparaison avec des modèles de l'état de l'art.

Classification des arguments

Nous effectuons également des comparaisons avec des études supposant que les arguments sont déjà connus. Nous comparons nos résultats dans le tableau 5.6 avec ceux de Zhang et al. (2021a) et Lin et al. (2023), qui proposent des approches de type zéro-shot (sans exemple). L'objectif de cette sous-tâche de classification des arguments est simplement d'attribuer un rôle à chaque argument, sachant qu'il a nécessairement un rôle dans l'événement. Cette approche élimine la nécessité de gérer la classe *NULLE*. Pour ces comparaisons, nous utilisons le modèle que nous avons présenté, mais sans le second terme de la fonction de coût qui ne concerne que la classe *NULLE* et sans l'étape de seuillage qui ne sert qu'à traiter la classe *NULLE*.

Zhang et al. (2021a) proposent pour leur part une approche prototypique dans laquelle ils construisent un prototype pour chaque rôle à partir de la sémantique de ce rôle. En pratique, pour chaque rôle donné, ils sélectionnent des phrases contenant l'expression de ce rôle dans le corpus NYT (Sandhaus, 2008) [7] pour enrichir les données. Ils encodent ensuite ces phrases de sorte à obtenir des représentations contextuelles de ces rôles, qu'ils agrègent pour former le prototype. Les arguments d'une mention sont ensuite classifiés en fonction de leur similarité à ces prototypes. Ils ajoutent ensuite une étape de filtrage de contraintes de conformité entre les types des entités en leurs rôles, qu'ils incorporent par programmation linéaire en nombres entiers (*Integer Linear Programtion*, ILP (Roth and Yih, 2004)). Nous comparons nos résultats avec cette approche dans deux configurations différentes : la version « 23-Types » est entraînée sur les 10 types d'événements les plus fréquents dans le jeu de données ACE 2005 et évaluée sur les 23 types restants tandis que la version « 33-Types » a été entraînée uniquement sur les données externes et évaluée sur les 33 types de ACE-2005.

Lin et al. (2023) mettent en œuvre une approche similaire à Sainz et al. (2022a), mais cette fois-ci par génération. Pour un type d'événement et un argument donnés, ils génèrent toutes les formulations possibles pour chaque rôle, puis évaluent le score de génération pour chacune des formulations. Par exemple, dans la phrase « *Alice hits Bob's head.* », la phrase « *Alice is the attacker* » aura un score de génération plus élevé que la phrase « *Alice is the victim* ». Le rôle ayant le score le plus élevé est sélectionné comme prédiction initiale, qui est ensuite filtrée à l'aide de contraintes de conformité entre type d'entité et rôle pour éviter les prédictions aberrantes. Cette approche évalue en quelque sorte la cohérence entre la phrase générée et le contexte, qui est la mention d'événement. Nous comparons notre modèle à leur modèle utilisant `GPT-J-6B`, qui est un modèle auto-régressif avec six milliards de paramètres, ainsi qu'à leur modèle utilisant `BERT-large-uncased`, qui est l'encodeur que nous utilisons. L'avantage principal de cette méthode réside dans le fait qu'elle ne nécessite ni données annotées, ni ressources externes pour fonctionner. Cependant, elle dépend d'un modèle de langue performant capable d'évaluer la cohérence entre deux phrases.

Lin et al. (2023)†		Zhang et al. (2021a)		**BERT-GCN**
GPT-J	BERT	23 types	33 types	
66,1	58,0	68,5	53,6	**79,5**

Table 5.6 – Résultats de la classification des arguments dans une configuration 5-shots. les résultats marqués † sont issus de l'article d'origine. Nos résultats sont en **gras**.

Le tableau 5.6 montre clairement que notre méthode surpasse de manière significative les méthodes de l'état de l'art. Ces résultats renforcent encore davantage la

7. Corpus NYT

pertinence de notre nouvelle formulation par méta-apprentissage et mettent en évidence son potentiel pour la classification des arguments d'événements à partir de peu d'exemples.

5.5 . Discussions

5.5.1 . Détail des performances par rôle

Nous donnons les résultats détaillés par rôle et par type d'événement dans le tableau 5.7. Par ailleurs, nous comparons nos trois encodeurs à la figure 5.9, où nous donnons la moyenne de la F1-mesure pour chaque rôle. Dans l'ensemble, ces résultats montrent que l'intérêt des informations syntaxiques est observé à la fois pour les rôles vus pendant l'entraînement et pour les nouveaux rôles apparaissant uniquement pendant l'évaluation [8]. Les apports se manifestent particulièrement pour les rôles pour lesquels l'encodeur **BERT** standard montre des performances relativement faibles. En revanche, pour les rôles considérés comme « faciles » [9], l'encodeur **BERT** standard demeure très compétitif et les informations syntaxiques ont un impact moins significatif, voire parfois négatif. En particulier, l'encodeur **BERT-GCN** semble contribuer principalement à équilibrer les performances pour les rôles qui peuvent être confondus, tels que *Origin* et *Destination* ou encore *Buyer* et *Seller*.

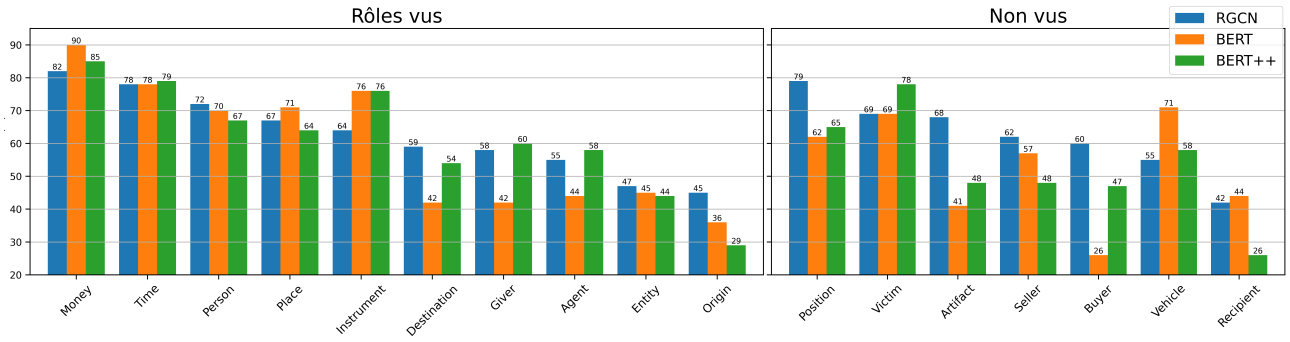


Figure 5.9 – Moyenne de la F1-mesure par rôle pour les trois types d'encodeurs **BERT**, **BERT++** et **BERT-GCN**. À gauche, nous avons les rôles vus pendant l'entraînement des modèles et à droite, les rôles vu seulement pendant l'évaluation.

La figure 5.10 donne pour sa part un aperçu des représentations élaborées par chaque encodeur sur l'ensemble d'évaluation. L'encodeur **BERT pré-entraîné** correspond à un modèle BERT sans aucun ajustement spécifique à la tâche d'extraction des arguments d'événements. Nous comparons les trois encodeurs présentés dans ce travail : **BERT**, **BERT++** et **BERT-GCN**.

8. Les rôles vus pendant l'entraînement sont les rôles qui portent le même nom, mais qui correspondent à des types d'événements différents.
9. Les rôles qui ne portent pas à confusion tels que *Instrument*, *Véhicule* ou *Money*.

Type	Sous-Type	Argument	Précision	Rappel	F1-mesure
Life	Be-Born	Person	0,67	0,90	0,77
Life	Die	Agent	0,59	0,59	0,59
Life	Die	Instrument	0,93	0,42	0,58
Life	Die	Place	0,70	0,66	0,68
Life	Die	Time	0,74	0,91	0,82
Life	Die	Victim	0,78	0,59	0,67
Life	Divorce	Person	0,88	0,81	0,84
Life	Injure	Agent	0,89	0,36	0,52
Life	Injure	Instrument	0,76	0,65	0,70
Life	Injure	Place	0,69	0,77	0,73
Life	Injure	Time	0,79	0,94	0,86
Life	Injure	Victim	0,75	0,67	0,71
Life	Marry	Person	0,88	0,49	0,63
Movement	Transport	Agent	0,54	0,57	0,55
Movement	Transport	Artifact	0,79	0,59	0,68
Movement	Transport	Destination	0,87	0,44	0,59
Movement	Transport	Origin	0,38	0,56	0,45
Movement	Transport	Time	0,66	0,65	0,66
Movement	Transport	Vehicle	0,68	0,46	0,55
Personnel	Elect	Entity	0,35	0,53	0,42
Personnel	Elect	Person	0,89	0,47	0,62
Personnel	Elect	Place	0,59	0,59	0,59
Personnel	Elect	Position	1,00	0,69	0,82
Personnel	Elect	Time	0,79	0,85	0,82
Personnel	End-Position	Entity	0,44	0,49	0,47
Personnel	End-Position	Person	0,75	0,59	0,66
Personnel	End-Position	Position	0,66	0,70	0,68
Personnel	End-Position	Time	0,64	0,76	0,70
Personnel	Start-Position	Entity	0,70	0,42	0,53
Personnel	Start-Position	Person	0,84	0,72	0,78
Personnel	Start-Position	Position	0,87	0,87	0,87
Personnel	Start-Position	Time	0,87	0,76	0,81
Transaction	Transfer-Money	Giver	0,64	0,53	0,58
Transaction	Transfer-Money	Money	0,93	0,74	0,82
Transaction	Transfer-Money	Recipient	0,66	0,31	0,42
Transaction	Transfer-Money	Time	0,82	0,75	0,78
Transaction	Transfer-Ownership	Artifact	0,88	0,55	0,68
Transaction	Transfer-Ownership	Buyer	0,69	0,53	0,60
Transaction	Transfer-Ownership	Seller	0,68	0,58	0,62

Table 5.7 – Détail des performances par argument et par type d'événements.

Tout d'abord, il est évident que l'entraînement du modèle BERT améliore considérablement la qualité visuelle des plongements par rapport à un BERT non affiné. Cela met en lumière la pertinence de la formulation que nous avons adoptée pour cette tâche et l'importance de l'affinage du modèle BERT dans ce contexte.

On peut également observer qualitativement que les deux encodeurs enrichis, **BERT++** et **BERT-GCN**, semblent fournir des représentations plus discriminantes que l'encodeur BERT d'origine. Ces observations correspondent aux résultats obtenus lors de l'évaluation, ce qui renforce la pertinence de l'enrichissement des représentations par des informations syntaxiques et suggère une amélioration globale de la performance du modèle.

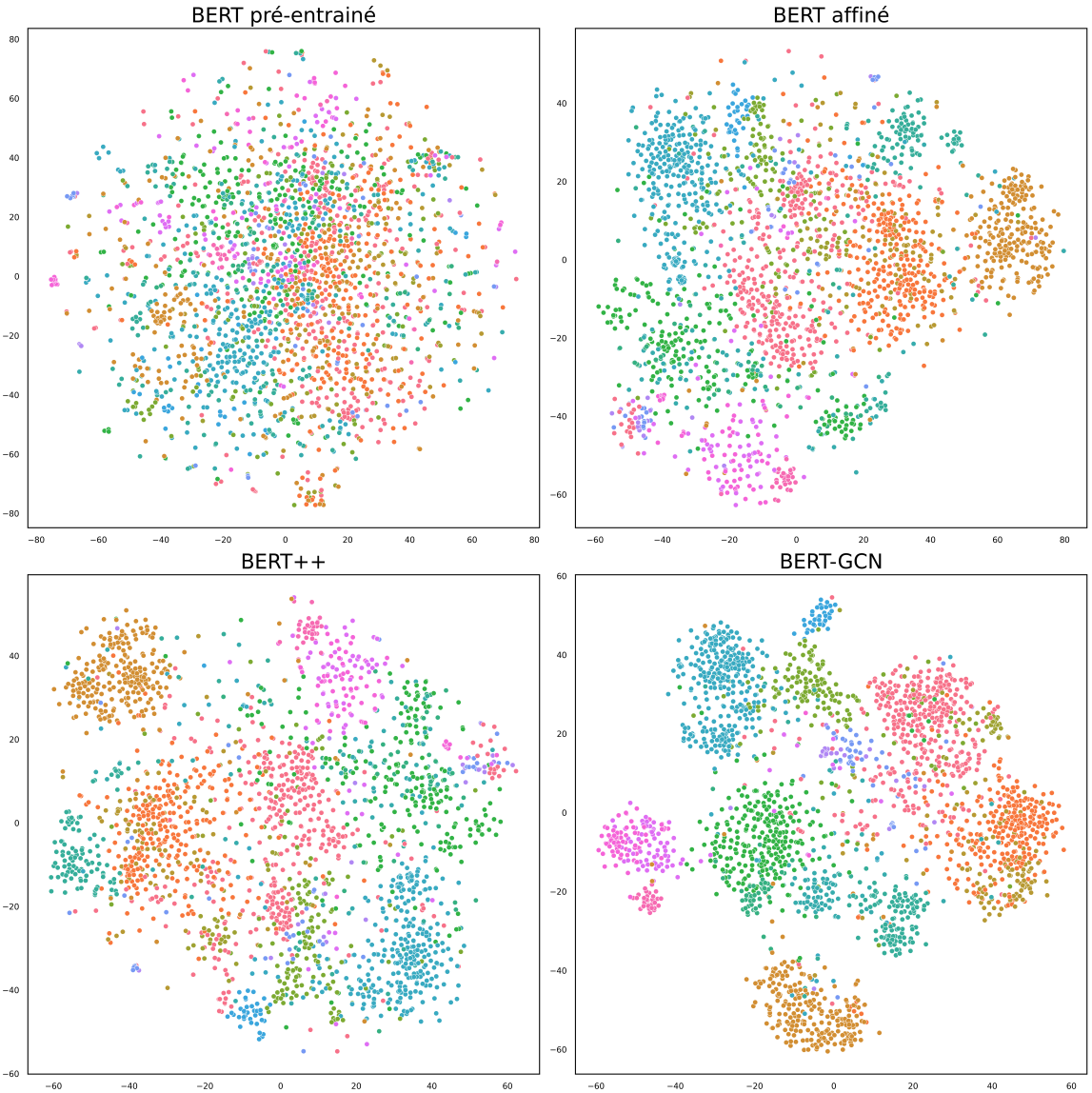


Figure 5.10 – Visualisation des représentations des arguments en utilisant la méthode t-SNE. Chaque point correspond à un argument et la couleur, à une classe de rôle.

5.5.2 . Études d'ablation

Nous présentons dans le tableau 5.8 une étude d'ablation réalisée avec le modèle **C-Proto**. L'objectif est d'explorer les effets de l'utilisation du seuillage et de l'introduction des contraintes liées au type des entités et à leurs rôles. Afin d'éliminer l'utilisation du seuil, nous avons reconstruit un prototype pour la classe nulle lors de la phase d'évaluation.

Les résultats obtenus dans cette étude d'ablation mettent en évidence l'utilité de chacune de ces opérations sur les performances globales du modèle. En général, nous observons que l'utilisation du seuillage conduit principalement à une amélioration de la précision, bien qu'elle soit accompagnée d'une légère diminution du rappel en contrepartie. Il faut noter que l'introduction du seuillage visait précisément à réduire le taux de faux positifs et donc, à une augmentation de la précision.

Les contraintes relatives aux types des entités et leurs rôles ont aussi un impact positif sur les performances, bien que leur contribution soit plus modeste par rapport au seuillage. Ces contraintes visent principalement à réduire les confusions entre certains arguments. Comme mentionné précédemment, il arrive que des entités de même type soient parfois confondues dans leurs rôles et, bien que ce filtrage n'élimine pas entièrement cette ambiguïté, il contribue néanmoins à la résoudre partiellement.

Ces contraintes de conformité ont un impact plus important avec l'encodeur **BERT++** par rapport aux deux autres encodeurs utilisés. Cela suggère que cette méthode a tendance à introduire plus de faux positifs au départ, que ces contraintes de conformité contribuent à éliminer. Il est donc possible que l'introduction des informations syntaxiques de cette manière puisse également entraîner des confusions, en particulier avec des entités qui ne sont pas des arguments, mais qui partagent des relations syntaxiques similaires avec les déclencheurs. En particulier, l'opération de *Max-pooling* sur le plus court chemin de dépendance entre l'entité et le déclencheur peut être à l'origine de ce phénomène. En effet, cette opération consiste à sélectionner la valeur maximale parmi les étiquettes de dépendance syntaxique composant ce plus court chemin. Cela peut conduire à une simplification excessive de l'information, car seules les connexions les plus fortes sont conservées.

En revanche, l'encodeur **BERT-GCN** semble fonctionner de manière plus satisfaisante, avec une amélioration de la précision de 10 points de pourcentage lorsque aucune opération de filtrage n'est appliquée (seuillage et contraintes de type). Cela met en évidence la supériorité de l'injection dynamique par rapport à l'injection statique, même si les écarts de performance ont tendance à diminuer lorsque l'on ajoute les autres composants du modèle.

Par ailleurs, nous notons que sans l'application du seuillage et des contraintes, l'encodeur **BERT-GCN** demeure légèrement supérieur à l'encodeur **BERT++**, qui lui-même surpasse l'encodeur **BERT**. Cela confirme un peu plus la supériorité de cette méthode par rapport aux deux autres.

	Encodeur	Précision	Rappel	F1-mesure
BERT	Modèle entier	**62,7**	**57,0**	**60,0**
	- seuillage	32,6 ($\downarrow 30,1$)	64,9 ($\uparrow 7,9$)	43,4($\downarrow 16,6$)
	- contraintes	60,9 ($\downarrow 1,8$)	53,6 ($\downarrow 3,4$)	57,0($\downarrow 3,0$)
	- seuillage & contraintes	31,2 ($\downarrow 31,5$)	61,5($\uparrow 4,5$)	41,4($\downarrow 18,6$)
BERT++	Modèle entier	**65,8**	**58,8**	**62,1**
	- seuillage	41,4($\downarrow 24,4$)	65,9 ($\uparrow 7,1$)	48,5($\downarrow 13,6$)
	- contraintes	58,0 ($\downarrow 7,8$)	50,2 ($\downarrow 8,6$)	53,4($\downarrow 8,7$)
	- seuillage & contraintes	31,9 ($\downarrow 33,9$)	62,1 ($\uparrow 3,3$)	42,2 ($\downarrow 19,9$)
BERT-GCN	Modèle entier	**68,5**	**59,2**	**63,5**
	- seuillage	47,9 ($\downarrow 20,6$)	59,3 ($\uparrow 0,1$)	52,9 ($\downarrow 10,6$)
	- contraintes	68,2 ($\downarrow 0,3$)	50,9 ($\downarrow 8,3$)	58,3 ($\downarrow 5,2$)
	- seuillage & contraintes	33,9 ($\downarrow 34,6$)	61,9($\uparrow 2,7$)	43,8($\downarrow 19,7$)

Table 5.8 – Étude d'ablation pour chaque composante du modèle C-Proto dans une configuration 5-shots. Précision (P), Rappel (R) et F1-mesure en moyenne sur cinq expérimentations. ($\uparrow$) et ($\downarrow$) indiquent l'écart du score par rapport au modèle entier pour chacun des encodeurs.

5.5.3 . Limitations

Notre approche repose sur la disponibilité d'informations sur les déclencheurs et les entités, ce qui peut limiter son applicabilité dans des scénarios dans lesquels éléments ne sont pas fournis ou annotés de manière explicite. Cependant, il existe un potentiel d'adaptation dans de tels scénarios en utilisant un système de détection d'événements et une méthode d'extraction d'entités candidates avant d'appliquer notre approche. Cela consisterait à extraire des entités potentielles à partir du texte, puis à utiliser ces candidats comme entrée pour notre méthode. Néanmoins, cela pourrait introduire du bruit et des erreurs supplémentaires, car les systèmes de reconnaissance d'entités ne sont pas parfaits.

Une autre limitation réside dans le fait que notre formulation se concentre uniquement sur les interactions entre les déclencheurs et les entités en négligeant les interactions entre les entités elles-mêmes. En effet, les interactions entre entités peuvent également être importantes pour l'extraction des arguments d'événements, comme le démontrent les travaux de Sha et al. (2018). Celles-ci pourraient être incorporées de façon similaire aux interactions entre les déclencheurs et les entités.

Par ailleurs, bien que les encodeurs améliorés **BERT++** et **BERT-GCN** affichent des performances supérieures à celles de l'encodeur **BERT** standard, ils introduisent également une augmentation du nombre de paramètres dans le modèle global. Il serait préférable d'effectuer une évaluation en ajustant le nombre de paramètres de chaque modèle afin de garantir une comparaison équitable. Cette question soulève un biais inhérent à la plupart des modèles de langue actuels, qui ne mettent pas systématique-

ment en rapport la taille des modèles et leurs performances. Nous pensons que la taille des modèles devrait être prise en compte de manière explicite dans les comparaisons entre modèles, en particulier dans un contexte d'apprentissage frugal où chaque petite amélioration peut avoir un impact considérable sur les performances des modèles. De plus, les RGCN peuvent être gourmands en ressources computationnelles, en particulier lorsque nous avons un grand nombre de relations différentes au sein des graphes.

Enfin, la place prépondérante des informations syntaxiques dans notre approche soulève un défi majeur : elle est soumise à la nécessité d'utiliser des outils d'analyse syntaxique adaptés, ce qui limite considérablement son application aux langues dépourvues de telles ressources linguistiques.

5.6 . Conclusion

Dans ce chapitre, nous avons exploré l'extraction des arguments d'événements dans un scénario de faible disponibilité de données pour définir de nouvelles classes, en mettant l'accent sur l'utilisation d'informations syntaxiques pour améliorer les performances. Nous avons repensé ce problème en le formulant comme une tâche de classification de relations entre le déclencheur et les arguments et en le traitant à l'aide de réseaux prototypiques. Dans ce contexte, nous avons adapté le cadre de classification N-ways, k-shots pour répondre aux besoins spécifiques de l'extraction d'événements à partir de peu d'exemples.

Nos objectifs dans ce travail étaient d'une part d'évaluer l'effectivité des propositions faites dans le chapitre 4, et d'autre part, examiner des méthodes d'enrichissement du modèle à l'aide d'informations syntaxiques. Nous avons exploré deux méthodes d'enrichissement, l'une statique, **BERT++**, et l'autre dynamique, **BERT-GCN**, à l'aide de réseaux convolutifs de graphes (GCN). Nos expériences ont révélé que l'intégration d'informations syntaxiques était bénéfique dans toutes les configurations et que l'approche **BERT-GCN** se démarquait lorsqu'il y avait très peu d'exemples d'entraînement disponibles.

Nos expérimentations sur le jeu de données ACE-2005 ont mis en lumière la robustesse et l'efficacité de ces propositions. Nous avons démontré que l'intégration d'informations syntaxiques dans les plongements des rôles d'événements pouvait notablement améliorer les performances. Ces résultats ouvrent de nouvelles perspectives pour l'amélioration de l'extraction d'arguments d'événements à partir de peu d'exemples dans une configuration où de nouveaux types d'événements apparaissent au cours du temps.

Pour les perspectives futures, nous envisageons de continuer à développer des modèles adaptés aux scénarios de faible disponibilité de données pour l'extraction d'événements. Cela pourrait inclure l'enrichissement des représentations de phrases en tenant compte des interactions entre les arguments ainsi que l'exploration d'approches fon-

dées sur des instructions pour renforcer la capacité des modèles à généraliser. Nous envisageons également de combiner notre approche avec un système d'extraction d'arguments candidats afin de pouvoir réaliser la tâche de bout-en-bout.

Chapitre 6

Conclusions et perspectives

Sommaire

6.1 . Bilans des contributions de la thèse

Ce travail de thèse a exploré divers aspects de l'extraction d'événements dans un contexte de données limitées avec l'utilisation de méthodes de méta-apprentissage. Nous avons abordé la tâche en adoptant une perspective axée sur l'adaptation au domaine qui considère l'émergence de nouvelles classes pour lesquelles seulement un petit ensemble de données annotées est disponible. Cette problématique est un axe de recherche qui anime de plus en plus la communauté des chercheurs en intelligence artificielle. Pour résoudre ce défi, nous nous sommes tournés vers une approche appelée le méta-apprentissage qui consiste à entraîner des modèles sur un grand nombre de tâches diverses de sorte qu'ils puissent s'adapter rapidement à de nouvelles tâches ayant peu de données annotées. Cette approche a montré des résultats prometteurs dans divers domaines, notamment la vision par ordinateur.

Dans ce contexte, les approches prototypiques, que nous avons utilisées dans ce travail, présentent plusieurs avantages essentiels. Elles sont non paramétriques et modulaires, simplifiant ainsi leur implémentation et leur adaptation. De plus, elles ne né-

cessitent pas d'entraîner à nouveau les modèles pour l'ajout, la suppression ou la modification de certaines classes dans l'ensemble d'évaluation. Cet agnosticisme vis-à-vis des classes d'évaluation rend ces méthodes particulièrement flexibles et permettent un déploiement efficace.

Plus précisément, nos contributions dans ce cadre se résument comme suit :

Dans le Chapitre 3, nous avons abordé la détection d'événements à partir de peu d'exemples en utilisant une approche de méta-apprentissage. Nous avons formulé cette tâche comme une tâche d'annotation de séquences et avons exploré différentes stratégies d'enrichissement des représentations vectorielles des mots fournies par le modèle de langue BERT (Devlin et al., 2019a). Les méthodes d'injection de connaissances et de combinaison de couches que nous avons proposées ont toutes deux conduit à des améliorations significatives par rapport à l'utilisation standard de BERT, permettant ainsi de surpasser les performances de l'état de l'art pour cette tâche.

Dans le Chapitre 4, nous nous sommes penché sur la gestion spécifique de la classe *NULLE* dans le cadre de la détection d'événements. La classe *NULLE* est une classe représentant souvent un défi pour cette tâche et pour plusieurs autres tâches d'extraction d'information. Cette classe spéciale se caractérise par la présence d'un ensemble hétérogène d'exemples qui ne sont pas directement rattachés à une classe d'intérêt pour la tâche. Nous avons examiné plusieurs stratégies pour résoudre ce problème, notamment la redéfinition du prototype de la classe NULLE et l'utilisation de méthodes de seuillage dynamique suite à un apprentissage contrastif. Nos expériences ont montré que les méthodes de seuillage dynamique étaient particulièrement efficaces. Cette nouvelle formulation de la détection d'événements ouvre de nouvelles perspectives pour la gestion de la classe NULLE dans d'autres tâches d'extraction d'information.

Dans le chapitre 5, nous nous sommes concentré sur l'extraction des arguments d'événements, toujours dans un scénario à faible disponibilité de données annotées, en mettant l'accent sur l'utilisation d'informations syntaxiques pour enrichir les plongements contextuels fournis par le modèle de langue BERT. Nous avons introduit une approche fondée sur le méta-apprentissage et avons exploré deux méthodes d'enrichissement des représentations de rôles d'événements, l'une statique, par concaténation et l'autre dynamique, utilisant les réseaux convolutifs de graphes (BERT-GCN). Nos résultats ont montré que l'intégration d'informations syntaxiques était bénéfique dans toutes les configurations et que l'approche BERT-GCN se démarquait en particulier lorsque très peu d'exemples étaient disponibles.

6.2 . Discussions

6.2.1 . Évaluation N ways, k shots

L'évaluation des modèles d'extraction d'événements à partir de peu d'exemples présente une grande variabilité d'une méthode à l'autre, ce qui se retrouve également dans plusieurs autres tâches de Traitement Automatique des Langues. La méthode d'évaluation elle-même devient une forme d'hyper-paramètre influençant directement les performances des modèles. Bien que le cadre de l'évaluation N-ways, k-shots soit souvent utilisé comme un cadre de référence dans les études académiques, il ne reflète pas nécessairement des cas d'usage réalistes. Cette méthode effectue un échantillonnage pour équilibrer le nombre de classes, ce qui ne correspond pas à la distribution réelle des classes dans le monde réel. De plus, elle évalue souvent les performances en calculant la moyenne sur un nombre fixe d'épisodes à partir de l'ensemble de test, plutôt que d'évaluer l'ensemble de test dans son intégralité. Cette évaluation dépend fortement des paramètres N, k, du nombre d'épisodes d'évaluation ($|\mathcal{E}_{test}|$), ainsi que de la qualité des ensembles de support. Ce problème suscite actuellement de nombreuses discussions concernant les performances réelles des modèles.

Plusieurs tentatives visant à rendre ces évaluations plus réalistes ont été menées. Par exemple, dans l'étude menée par Yang and Katiyar (2020) en 2020, l'échantillonnage est effectué uniquement sur le support set et les modèles sont évalués sur l'ensemble complet de l'ensemble d'évaluation, plutôt que de façon épisodique. C'est précisément cette approche que nous avons adoptée pour nos évaluations des chapitres 4 et 5.

En 2021, Bragg et al. (2021) ont introduit l'outil d'évaluation FLEX dans le but de créer un cadre d'évaluation unifié pour 20 tâches de traitement automatique du langage naturel nécessitant peu d'exemples d'apprentissage [1]. L'objectif principal était de rendre les propositions de recherche directement comparables entre elles. Pour atteindre cet objectif, FLEX a mis en place deux tableaux de classement (*leaderboards*) [2], bien que leur utilisation semble encore limitée à ce jour. En plus de fournir un code d'évaluation standard et les tableaux de classement, FLEX a introduit un cadre d'évaluation flexible permettant de faire varier les paramètres tels que N, k et le nombre d'épisodes d'évaluation. Le scénario de N variable, en particulier, correspond à notre étude sur l'extraction d'arguments, où le nombre d'arguments, et donc de classes, pouvait varier en fonction du type d'événement. Le paramètre k variable, quant à lui, reflète la possibilité que certaines classes puissent disposer de plus d'exemples d'apprentissage que d'autres, même si le nombre total d'exemples reste relativement faible pour l'ensemble des classes. Cette flexibilité contribue à une évaluation plus réaliste et adaptée à la diversité des problèmes abordés.

Plus récemment, Ma et al. (2023c) ont proposé un cadre d'évaluation unifié pour comparer les modèles de détection d'événements, mettant particulièrement l'accent sur la comparaison entre les méthodes génératives par invites (*prompting*) et les méthodes prototypiques. Leurs résultats diffèrent significativement des performances ini-

1. Ces tâches ne comprennent pas l'extraction d'événements.
2. FLEX et FLEX-Meta

tialement rapportées dans les articles d'origine. Cependant, ils concluent que, dans l'ensemble, les méthodes prototypiques surpassent nettement les méthodes génératives, y compris ChatGPT, en termes de performances. Cette constatation souligne l'importance de continuer à explorer et à développer des approches prototypiques pour l'extraction d'événements à partir de peu d'exemples.

En résumé, cette diversité dans les méthodes d'évaluation crée des biais potentiels dans les évaluations des modèles. Cela rend complexe la mesure objective et réaliste des performances, suscitant ainsi des débats sur la validité des résultats.

6.2.2 . Utilisation dans des applications réelles

Si les recherches en apprentissage à partir de peu d'exemples visent des applications dans le monde réel, elles ne satisfont pas encore pleinement ces exigences. Les applications réelles peuvent impliquer divers autres défis, tels que la distinction entre des classes très similaires (*fine-grained*), la présence de classes absentes du support set (reconnaissance en open set), des déséquilibres importants entre les classes, la combinaison de classes avec peu de données et de classes bien dotées, ou encore une grande variabilité au sein des exemples d'une même classe. Ces problématiques ont été explorées dans la thèse de M. Étienne Bennequin, en particulier dans son article (Bennequin et al., 2022), où il met en évidence les limites de ces méthodes dans des contextes d'applications réelles.

Par ailleurs, malgré plusieurs décennies d'études sur la tâche d'extraction d'événements et des performances de plus en plus élevées, son application pratique dans des applications réelles demeure limitée. Les meilleurs résultats actuels atteignent environ 80% pour la détection des déclencheurs et 60% pour les arguments. Comparativement à d'autres tâches telles que la traduction automatique ou l'analyse des sentiments, l'application de l'extraction d'événements dans des scénarios réels demeure en deçà des attentes, en particulier dans des contextes avec peu d'exemples, où les performances sont encore plus modestes et les méthodes d'évaluation moins rigoureuses.

6.2.3 . Quand la taille compte

La taille des modèles encodeurs peut jouer un rôle critique dans les performances, notamment dans le contexte de l'apprentissage à partir de peu d'exemples. Les modèles plus volumineux ont tendance à mieux généraliser grâce à leur capacité à capturer davantage d'informations et à apprendre des détails plus fins.

Une des conséquences de cette tendance est que les gains de performance deviennent moins attribuables au modèle en lui-même. Lorsque tous les modèles sont de grande taille, les variations dans le traitement des données, les architectures ou les approches peuvent être noyées dans la masse du modèle de langue. Il devient alors difficile de mettre en évidence les avantages comparatifs de différentes approches, no-

tamment dans le cadre des réseaux prototypiques où les poids de l'encodeur ne sont pas mis à jour. Cette question a été examinée dans l'étude de Dopierre et al. (2021), qui compare différentes méthodes prototypiques pour la classification de textes à partir de peu d'exemples. Leurs résultats montrent que les méthodes les plus récentes et supposément meilleures ne fournissent parfois pas de meilleures performances par rapport aux méthodes plus anciennes et plus simples lorsqu'elles ont toutes accès au même encodeur. De plus, ils ont démontré qu'un réseau prototypique simple est souvent plus robuste et performant que des variantes plus complexes.

Par ailleurs, l'utilisation de modèles de plus en plus volumineux soulève des préoccupations en matière de ressources informatiques, de coûts énergétiques et d'impact environnemental. L'entraînement et le déploiement de ces modèles géants exigent des infrastructures de calcul puissantes, ce qui n'est pas à la portée de tous, limitant ainsi la reproductibilité des recherches. Ces considérations suscitent des inquiétudes en matière de durabilité et d'éthique scientifique. Nous pensons qu'il est essentiel de prendre en compte les besoins spécifiques des tâches, les contraintes en ressources et les considérations environnementales lors du choix des modèles. De plus, il est crucial de comparer les modèles en tenant compte explicitement de leurs différences en termes de taille et de processus de pré-entraînement. Ces modèles excellent dans des scénarios à faible volume de données grâce à leur pré-entraînement sur de vastes ensembles de données, ce qui leur confère une capacité de généralisation exceptionnelle. Il est regrettable que cela ne soit pas pris en compte de manière adéquate et explicite lors des évaluations.

6.3 . Qu'en-est-il des « gros » modèles génératifs (*Large Language Models, LLMs*) ?

La période récente dans le domaine de l'apprentissage automatique est marquée par la montée en puissance des modèles de langue pré-entraînés, en particulier les géants tels que GPT-3 (Brown et al., 2020), BLOOM (Scao et al., 2022) ou encore OPT (Zhang et al., 2022e). Ces modèles ont atteint des dimensions impressionnantes, avec des centaines de milliards de paramètres, ouvrant de nouvelles perspectives pour de nombreuses tâches du TAL. Leur capacité à s'adapter rapidement à de nouvelles tâches sans nécessiter d'apprentissage préalable les rend particulièrement intéressants pour les problématiques avec peu de données annotées. D'une part, ils peuvent s'adapter rapidement à de nouvelles tâches sans nécessiter d'apprentissage préalable (*Zero Shot*). D'autre part, ils peuvent générer de nouveaux exemples annotés nécessaires à l'apprentissage de modèles plus petits. Enfin, leur capacité à travailler avec plusieurs langues facilite l'adaptation des tâches à de nouvelles langues.

Par ailleurs, l'utilisation de modèles comme ChatGPT peut offrir une certaine forme d'explicabilité, notamment parce que ces modèles sont capables de générer des expli-

cations dans les prédictions qu'ils font. Un autre avantage peut être l'extraction en domaine ouvert, c'est-à-dire lorsque les types d'événements ne sont pas fixes et que l'on souhaite en découvrir de nouveaux.

Plusieurs évaluations de ChatGPT ont été menées depuis sa sortie en novembre 2022 pour évaluer ses capacités sur plusieurs tâches d'extraction d'information (Gao et al., 2023; Wei et al., 2023; Li et al., 2023). Les résultats de Li et al. (2023) sont donnés dans le Tableau 6.1. En dehors de la tâche de typage d'entités (*Entity Typing*), qui consiste simplement à assigner un type prédéfini à des entités, ChatGPT reste assez mauvais sur les tâches d'extraction d'information comparé à l'état de l'art et à un modèle de référence fondé sur BERT. Cependant, il faut reconnaître qu'il est évalué sans aucun entraînement alors que les modèles de l'état de l'art sont affinés sur les tâches cibles. Dans les tâches complexes telles que l'extraction de relations ou d'événements, ChatGPT éprouve des difficultés, car il doit d'abord identifier les entités qui participent à ces relations ou événements, puis leur associer des types, ce qui demande de comprendre tout le contexte global des phrases ou des documents en entrée. La complexité de telles tâches demanderait sans doute de structurer les invites faites à ces modèles pour expliciter les différentes sous-tâches à réaliser dans des approches de type *Chain-of-Throught* (Wei et al., 2022).

	Dataset	BERT	SOTA	ChatGPT
Entity Typing(ET)	**BBN**	80,3	82,2 (Zuo et al., 2022)	85,6
	OntoNotes 5.0	69,1	72,1 (Zuo et al., 2022)	73,4
Named Entity Recognition(NER)	**CoNLL2003**	92,8	94,6 (Wang et al., 2021b)	67,2
	OntoNotes 5.0	89,2	91,9 (Ye et al., 2022)	51,1
Relation Classification(RC)	**TACRED**	72,7	75,6 (Li et al., 2022a)	20,3
	SemEval2010	89,1	91,3 (Zhao et al., 2021)	42,5
Relation Extraction(RE)	**ACE05-R**	87,5 | 63,7	91,1 | 73,0 (Ye et al., 2022)	40,5 | 4,5
	SciERC	65,4 | 43,0	69,9 | 53,2 (Ye et al., 2022)	25,9 | 5,5
Event Detection(ED)	**ACE05-E**	71,8	75,8 (Liu et al., 2022a)	17,1
	ACE05-E+	72,4	72,8 (Lin et al., 2020)	15,5
Event Argument Extraction(EAE)	**ACE05-E**	65,3	73,5 (Hsu et al., 2022)	28,9
	ACE05-E+	64,0	73,0 (Hsu et al., 2022)	30,9
Event Extraction(EE)	**ACE05-E**	71,8 | 51,0	74,7 | 56,8 (Lin et al., 2020)	17,0 | 7,3
	ACE05-E+	72,4 | 52,7	71,7 | 56,8 (Hsu et al., 2022)	16,6 | 7,8

Table 6.1 – Évaluation de ChatGPT sur les tâches d'extraction d'informations. Comparaison avec l'état de l'art sur 7 tâches d'extraction d'informations sur 8 jeux de données. Source : Li et al. (2023).

Une limite commune à toutes ces évaluations est que ces modèles dépendent énormément des instructions utilisées pour les évaluer. Cela rend donc les recherches moins intéressantes dans ce sens.

6.4 . Perspectives

Ce travail apporte des contributions significatives à l'étude de l'extraction d'événements à partir de données limitées dans une perspective d'adaptation au domaine, ouvrant la voie à de nouvelles avancées potentielles dans ce domaine. Pour les perspectives de la thèse, il reste encore de nombreuses pistes à explorer. Parmi elles, on peut citer l'intégration de différentes sources de connaissances pour enrichir les modèles, l'extension des méthodes proposées à d'autres tâches d'extraction d'information et à d'autres domaines [3] et l'exploration d'approches génératives pour améliorer la performance des modèles.

Ces améliorations potentielles peuvent être regroupées selon quatre axes principaux.

Amélioration des approches proposées : Les modèles développés dans cette thèse, qu'il s'agisse de la détection d'événements, de la gestion de la classe *NULLE* ou de l'extraction d'arguments d'événements, peuvent tous bénéficier de travaux d'amélioration continue. Des techniques d'entraînement plus avancées, des architectures de modèle plus sophistiquées ou des méthodes d'enrichissement des encodeurs plus efficaces pourraient être explorées pour augmenter les performances. De plus, dans le cas spécifique de l'extraction des arguments d'événements, nous nous appuyons sur la connaissance des entités issues des jeux de données, ce qui constitue une hypothèse non réaliste. Une amélioration potentielle dans ce contexte consisterait à explorer comment ces méthodes fonctionnent lorsque nous n'avons pas accès à ces entités et lorsque celles-ci doivent être détectées automatiquement.

Application à d'autres tâches : Les méthodologies développées dans cette thèse ne sont pas limitées aux tâches de traitement d'événements. Elles pourraient être étendues à d'autres tâches de l'extraction d'information, telles que la reconnaissance d'entités nommées, l'extraction de relations et même d'autres tâches de TAL.

L'adaptation au domaine : L'application de ces méthodes à l'adaptation au domaine constitue également une perspective importante. Elle pourrait être particulièrement utile dans des scénarios où l'extraction d'événements doit être adaptée à des domaines spécifiques, tels que la médecine ou le droit, domaines dans lesquels les données annotées sont souvent rares en raison de problèmes de confidentialité ou de la nécessité d'une expertise particulière pour les annoter. De plus, les méthodes proposées dans notre thèse pourraient également être utilisées pour l'adaptation à de nouvelles langues, étendant ainsi la portée de ces modèles à un éventail plus large de contextes linguistiques.

3. Cela peut concerner le domaine applicatif, la langue ou les sources des données.

Pour plus de frugalité : Le travail réalisé dans le cadre de cette thèse a porté une attention particulière à l'utilisation de ressources limitées en termes de données annotées. Les recherches futures pourraient se pencher sur d'autres aspects de la frugalité en apprentissage profond, tels que la réduction de la taille des modèles, l'optimisation des temps de calcul, de la mémoire ou de la consommation énergétique dans les applications d'apprentissage automatique, en particulier dans des situations où les données sont peu disponibles.

Bibliographie

Jacqueline Aguilar, Charley Beller, Paul McNamee, Benjamin Van Durme, Stephanie Strassel, Zhiyi Song, and Joe Ellis. 2014. A comparison of the events and relations across ACE, ERE, TAC-KBP, and FrameNet annotation standards. In *Proceedings of the Second Workshop on EVENTS : Definition, Detection, Coreference, and Representation*, pages 45–53, Baltimore, Maryland, USA. Association for Computational Linguistics. 12

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 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*, CIKM '19, page 1823–1832, New York, NY, USA. Association for Computing Machinery. 53

Douglas E. Appelt and Boyan Onyshkevych. 1998. The common pattern specification language. In *TIPSTER TEXT PROGRAM PHASE III : Proceedings of a Workshop held at Baltimore, Maryland, October 13-15, 1998*, pages 23–30, Baltimore, Maryland, USA. Association for Computational Linguistics. 4

Jun Araki and Teruko Mitamura. 2015. Joint event trigger identification and event coreference resolution with structured perceptron. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2074–2080, Lisbon, Portugal. Association for Computational Linguistics. 13, 22

Collin F. Baker. 2014. FrameNet: A knowledge base for natural language processing. In *Proceedings of Frame Semantics in NLP : A Workshop in Honor of Chuck Fillmore (1929-2014)*, pages 1–5, Baltimore, MD, USA. Association for Computational Linguistics. 26

S. Baker and T. Kanade. 2000. Hallucinating faces. In *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, pages 83–88. 35

Ali Balali, Masoud Asadpour, Ricardo Campos, and Adam Jatowt. 2020. Joint event extraction along shortest dependency paths using graph convolutional networks. *Knowledge-Based Systems*, 210 :106492. 105

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics. 103, 107

E. Bennequin, M. Tami, A. Toubhans, and C. Hudelot. 2022. Few-shot image classification benchmarks are too far from reality: Build back better with semantic task sampling. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4766–4775, Los Alamitos, CA, USA. IEEE Computer Society. 126

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics. 163

Jari Björne and Tapio Salakoski. 2018. Biomedical event extraction using convolutional neural networks and dependency parsing. In *Proceedings of the BioNLP 2018 workshop*, pages 98–108, Melbourne, Australia. Association for Computational Linguistics. 14

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery. 12, 161

Emanuela Boros, Jose G. Moreno, and Antoine Doucet. 2022. Exploring entities in event detection as question answering. In *Advances in Information Retrieval*, pages 65–79, Cham. Springer International Publishing. 33, 34

Jonathan Bragg, Arman Cohan, Kyle Lo, and Iz Beltagy. 2021. Flex: Unifying evaluation for few-shot nlp. In *Neural Information Processing Systems*. 47, 125

Ofer Bronstein, Ido Dagan, Qi Li, Heng Ji, and Anette Frank. 2015. Seed-Based Event Trigger Labeling: How far can event descriptions get us? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2 : Short Papers)*, pages 372–376, Beijing, China. Association for Computational Linguistics. 30, 32, 34, 35, 42, 57, 64

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc. 19, 29, 38, 127

Mary Elaine Califf and Raymond J. Mooney. 1997. Relational learning of pattern-match rules for information extraction. In *CoNLL97 : Computational Natural Language Learning*. 14

Mary Elaine Califf and Raymond J. Mooney. 2003. Bottom-up relational learning of pattern matching rules for information extraction. *J. Mach. Learn. Res.*, 4(null) :177–210. 14

Ming-Wei Chang, Lev Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation : Dataless classification. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, AAAI'08, page 830–835. AAAI Press. 25

Danqi Chen. 2018. *Neural Reading Comprehension and Beyond*. Ph.D. thesis, Stanford University. 18

Daoyuan Chen, Yaliang Li, Minghui Qiu, Zhen Wang, Bofang Li, Bolin Ding, Hongbo Deng, Jun Huang, Wei Lin, and Jingren Zhou. 2021a. Adabert : Task-adaptive bert compression with differentiable neural architecture search. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20. 171

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2021b. Honey or Poison? Solving the Trigger Curse in Few-shot Event Detection via Causal Intervention. *arXiv :2109.05747 [cs]*. ArXiv : 2109.05747. 30, 34, 42, 64, 86, 92, 161

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. 2019. A closer look at few-shot classification. In *International Conference on Learning Representations*. 46

Yubo Chen, Shulin Liu, Shizhu He, Kang Liu, and Jun Zhao. 2016. Event extraction via bidirectional long short-term memory tensor neural networks. In *China National Conference on Chinese Computational Linguistics*. 16, 22, 24

Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. Automatically labeled data generation for large scale event extraction. In *Proceedings of the 55th Annual*

Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers), pages 409–419, Vancouver, Canada. Association for Computational Linguistics. 12, 22, 24, 26, 161

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015a. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics. 14, 15, 22, 24

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015b. Event Extraction via Dynamic Multi-Pooling Convolutional Neural Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics. 15, 105

Yunmo Chen, Tongfei Chen, Seth Ebner, Aaron Steven White, and Benjamin Van Durme. 2020. Reading the manual: Event extraction as definition comprehension. In *Proceedings of the Fourth Workshop on Structured Prediction for NLP*, pages 74–83, Online. Association for Computational Linguistics. 22, 24, 35, 100

Laura Chiticariu, Yunyao Li, and Frederick R. Reiss. 2013. Rule-based information extraction is dead! long live rule-based information extraction systems! In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 827–832, Seattle, Washington, USA. Association for Computational Linguistics. 4

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics. 166

Xin Cong, Shiyao Cui, Bowen Yu, Tingwen Liu, Wang Yubin, and Bin Wang. 2021. Few-Shot Event Detection with Prototypical Amortized Conditional Random Field. In *Findings of the Association for Computational Linguistics : ACL-IJCNLP 2021*, pages 28–40, Online. Association for Computational Linguistics. xi, 12, 28, 31, 34, 42, 45, 49, 51, 53, 55, 56, 67, 71, 72, 80, 91, 92

Shiyao Cui, Bowen Yu, Tingwen Liu, Zhenyu Zhang, Xuebin Wang, and Jinqiao Shi. 2020. Edge-Enhanced Graph Convolution Networks for Event Detection with Syntactic Relation. In *Findings of the Association for Computational Linguistics : EMNLP 2020*, pages 2329–2339, Online. Association for Computational Linguistics. 14, 17, 22, 24, 51

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: an architecture for development of robust HLT applications. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 168–175, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics. 4

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg. Springer Berlin Heidelberg. 113

Lu Dai, Bang Wang, Wei Xiang, and Yijun Mo. 2022. Bi-directional iterative prompt-tuning for event argument extraction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6251–6263, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 18, 35, 100, 103, 113

Shumin Deng, Ningyu Zhang, Jiaojian Kang, Yichi Zhang, Wei Zhang, and Huajun Chen. 2020. Meta-Learning with Dynamic-Memory-Based Prototypical Network for Few-Shot Event Detection. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 151–159, Houston TX USA. ACM. 12, 28, 31, 34, 43, 45, 49, 67, 78, 85, 161, 163

Shumin Deng, Ningyu Zhang, Luoqiu Li, Chen Hui, Tou Huaixiao, Mosha Chen, Fei Huang, and Huajun Chen. 2021. OntoED: Low-resource Event Detection with Ontology Embedding. 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 2828–2839, Online. Association for Computational Linguistics. 32, 34, 42, 78

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv :1810.04805 [cs]*. ArXiv : 1810.04805. 7, 16, 29, 38, 48, 53, 86, 124

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. 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. 105

Ning Ding, Xiaobin Wang, Yao Fu, Guangwei Xu, Rui Wang, Pengjun Xie, Ying Shen, Fei Huang, Haitao Zheng, and Rui Zhang. 2021. Prototypical representation learning for relation extraction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. 46

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) program – tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA). 2

Chenhe Dong, Guangrun Wang, Hang Xu, Jiefeng Peng, Xiaozhe Ren, and Xiaodan Liang. 2021. EfficientBERT: Progressively searching multilayer perceptron via warm-up knowledge distillation. In *Findings of the Association for Computational Linguistics : EMNLP 2021*, pages 1424–1437, Punta Cana, Dominican Republic. Association for Computational Linguistics. 171

Thomas Dopierre, Christophe Gravier, and Wilfried Logerais. 2021. A neural few-shot text classification reality check. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics : Main Volume*, pages 935–943, Online. Association for Computational Linguistics. 51, 127

Rotem Dror, Segev Shlomov, and Roi Reichart. 2019. Deep Dominance - How to Properly Compare Deep Neural Models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2773–2785, Florence, Italy. Association for Computational Linguistics. 54, 56

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics. 18, 22, 24, 32, 33, 35

Sanghamitra Dutta, Liang Ma, Tanay Kumar Saha, Di Liu, Joel Tetreault, and Alejandro Jaimes. 2021. GTN-ED: Event detection using graph transformer networks. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15)*, pages 132–137, Mexico City, Mexico. Association for Computational Linguistics. 17

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077, Online. Association for Computational Linguistics. 163

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2019. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55) :1–21. 171

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books. 28

Rui Feng, Jie Yuan, and Chao Zhang. 2020. Probing and fine-tuning reading comprehension models for few-shot event extraction. *ArXiv*, abs/2010.11325. 33, 34

Zhaoyang Feng, Xing Wang, and Deqian Fu. 2022. Dual machine reading comprehension for event extraction. In *2022 12th International Conference on Information Science and Technology (ICIST)*, pages 317–324. 18, 32

James Ferguson, Colin Lockard, Daniel Weld, and Hannaneh Hajishirzi. 2018. Semi-supervised event extraction with paraphrase clusters. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 2 (Short Papers)*, pages 359–364, New Orleans, Louisiana. Association for Computational Linguistics. 26

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017a. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *arXiv :1703.03400 [cs]*. ArXiv : 1703.03400. 5, 170

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017b. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1126–1135. JMLR.org. 33

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014. A discriminative graph-based parser for the Abstract Meaning Representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 1426–1436, Baltimore, Maryland. Association for Computational Linguistics. 36

Dayne Freitag and Nicholas Kushmerick. 2000. Boosted wrapper induction. In *AAAI/IAAI*. 14

Dayne Freitag and Andrew McCallum. 1999. Information extraction with hmms and shrinkage. Technical report, AAAI. 14

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2018. Few-shot classification in Named Entity Recognition Task. *arXiv :1812.06158 [cs, stat]*. ArXiv : 1812.06158. 5, 45, 74, 75, 99

Jun Gao, Changlong Yu, Wei Wang, Huan Zhao, and Ruifeng Xu. 2022. Mask-then-fill: A flexible and effective data augmentation framework for event extraction. In *Findings of the Association for Computational Linguistics : EMNLP 2022*, pages 4537–4544, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 26

Jun Gao, Huan Zhao, Changlong Yu, and Ruifeng Xu. 2023. Exploring the feasibility of chatgpt for event extraction. 128

Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019a. Hybrid Attention-Based Prototypical Networks for Noisy Few-Shot Relation Classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33 :6407–6414. 99, 103

Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019b. FewRel 2.0: Towards more challenging few-shot relation classification. 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 6250–6255, Hong Kong, China. Association for Computational Linguistics. 45, 74

Ruiying Geng, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. 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 3904–3913, Hong Kong, China. Association for Computational Linguistics. 31, 171

Reza Ghaeini, Xiaoli Fern, Liang Huang, and Prasad Tadepalli. 2016. Event nugget detection with forward-backward recurrent neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, pages 369–373, Berlin, Germany. Association for Computational Linguistics. 16

Ralph Grishman. 2019. Twenty-five years of information extraction. *Natural Language Engineering*, 25(6) :677–692. 1

Ralph Grishman and Beth Sundheim. 1996. Message Understanding Conference- 6: A brief history. In *COLING 1996 Volume 1 : The 16th International Conference on Computational Linguistics*. 11

Ralph Grishman, David Westbrook, and Adam Meyers. 2005. Nyu's english ace 2005 system description. 10

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics. 5, 103, 107

Li He, Xiya Zhao, Liang Zhao, and Qing Zhang. 2023. An event extraction approach based on a multi-round q&a framework. *Applied Sciences*, 13 :6308. 18

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 507–517, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee. 79

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531. 28

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9 :1735–80. 166

Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *Similarity-Based Pattern Recognition*, pages 84–92, Cham. Springer International Publishing. 46

Yu Hong, Wenxuan Zhou, Jingli Zhang, Guodong Zhou, and Qiaoming Zhu. 2018. Self-regulation: Employing a generative adversarial network to improve event detection. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 515–526, Melbourne, Australia. Association for Computational Linguistics. 22, 24

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1381–1393, Online. Association for Computational Linguistics. 55, 61

I-Hung Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. 2022. DEGREE: A data-efficient generation-based event extraction model. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 1890–1908, Seattle, United States. Association for Computational Linguistics. 19, 22, 24, 32, 33, 34, 35, 100, 128

Hongwei Huang, Zhangkai Wu, Wenbin li, Jing Huo, and Yang Gao. 2021. Local descriptor-based multi-prototype network for few-shot learning. *Pattern Recognition*, 116 :107935. 76

Lifu Huang, Taylor Cassidy, Xiaocheng Feng, Heng Ji, Clare R. Voss, Jiawei Han, and Avirup Sil. 2016. Liberal event extraction and event schema induction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 258–268, Berlin, Germany. Association for Computational Linguistics. 26

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-shot transfer learning for event extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 2160–2170, Melbourne, Australia. Association for Computational Linguistics. 34, 36, 100

Peixin Huang, Xiang Zhao, Ryuichi Takanobu, Zhen Tan, and Weidong Xiao. 2020. Joint event extraction with hierarchical policy network. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2653–2664, Barcelona, Spain (Online). International Committee on Computational Linguistics. 22, 24

Zhong Ji, Xingliang Chai, Yunlong Yu, Yanwei Pang, and Zhongfei Zhang. 2020. Improved prototypical networks for few-shot learning. *Pattern Recognition Letters*, 140 :81–87. 46

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12). 35

Thomas N. Kipf and Max Welling. 2017a. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. 16, 167

Thomas N. Kipf and Max Welling. 2017b. Semi-supervised classification with graph convolutional networks. 106

Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. 2015. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille. 46

Dorian Kodelja, Romaric Besançon, and Olivier Ferret. 2019. Exploiting a more global context for event detection through bootstrapping. In *Advances in Information Retrieval*, pages 763–770, Cham. Springer International Publishing. 15

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields : Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML'01)*, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. 55, 91

Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2021a. Learning Prototype Representations Across Few-Shot Tasks for Event Detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5270–5277, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics. 12, 28, 31, 34, 53, 67, 78, 99

Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2021b. Learning prototype representations across few-shot tasks for event detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5270–5277, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics. 109

Viet Dac Lai and Thien Nguyen. 2019. Extending Event Detection to New Types with Learning from Keywords. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 243–248, Hong Kong, China. Association for Computational Linguistics. 30, 42, 57, 78, 159

Viet Dac Lai, Thien Huu Nguyen, and Frank Dernoncourt. 2020a. Extensively Matching for Few-shot Learning Event Detection. In *Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events*, pages 38–45, Online. Association for Computational Linguistics. 32, 67, 78, 85

Viet Dac Lai, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020b. Event detection: Gate diversity and syntactic importance scores for graph convolution neural networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5405–5411, Online. Association for Computational Linguistics. 17

Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. 2015. Human-level concept learning through probabilistic program induction. *Science*, 350(6266) :1332–1338. 42

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In Armand Prieditis and Stuart Russell, editors, *Machine Learning Proceedings 1995*, pages 331–339. Morgan Kaufmann, San Francisco (CA). 25

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4) :541–551. 165

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11) :2278–2324. 167

Roger Levy and Galen Andrew. 2006. Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA). 4

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 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. 19, 35, 38, 113

Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023. Evaluating chatgpt's information extraction capabilities : An assessment of performance, explainability, calibration, and faithfulness. 30, 39, 128

Bo Li, Wei Ye, Jinglei Zhang, and Shikun Zhang. 2022a. Reviewing labels : Label graph network with top-k prediction set for relation extraction. 128

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. 2018. Learning to generalize : Meta-learning for domain generalization. In *AAAI Conference on Artificial Intelligence*. 170

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020a. Event extraction as multi-turn question answering. In *Findings of the Association for Computational Linguistics : EMNLP 2020*, pages 829–838, Online. Association for Computational Linguistics. 18, 22, 24

L. Li, Y. Liu, and M. Qin. 2020b. Extracting biomedical events with parallel multi-pooling convolutional neural networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17(02) :599–607. 15

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics. 13, 22, 24

Sha Li, Heng Ji, and Jiawei Han. 2021. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 894–908, Online. Association for Computational Linguistics. 19, 163

Sha Li, Liyuan Liu, Yiqing Xie, Heng Ji, and Jiawei Han. 2022b. P4e: Few-shot event detection as prompt-guided identification and localization. *ArXiv*. 32, 34

Wei Li, Dezhi Cheng, Lei He, Yuanzhuo Wang, and Xiaolong Jin. 2019a. Joint event extraction based on hierarchical event schemas from framenet. *IEEE Access*, 7 :25001–25015. 22, 24

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019b. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1340–1350, Florence, Italy. Association for Computational Linguistics. 18

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics. 16, 22, 24, 128

Zizheng Lin, Hongming Zhang, and Yangqiu Song. 2023. Global constraints with prompting for zero-shot event argument classification. In *Findings of the Association for Computational Linguistics : EACL 2023*, pages 2527–2538, Dubrovnik, Croatia. Association for Computational Linguistics. 36, 37, 100, 103, 108, 114, 115

Jian Liu, Yubo Chen, and Kang Liu. 2019. Exploiting the ground-truth: An adversarial imitation based knowledge distillation approach for event detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33 :6754–6761. 28

Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651, Online. Association for Computational Linguistics. 18, 32, 33, 34

Jian Liu, Yufeng Chen, and Jinan Xu. 2022a. Saliency as evidence: Event detection with trigger saliency attribution. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 4573–4585, Dublin, Ireland. Association for Computational Linguistics. 128

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9). 21

Qi Liu, Zhen Luan, Kunlong Wang, Ye Zou, Bing Liu, and Yang Zhang. 2023b. Document-level event extraction - a survey of methods and applications. *Journal of Physics : Conference Series*, 2504(1) :012008. 163

Shaobo Liu, Rui Cheng, Xiaoming Yu, and Xueqi Cheng. 2018a. Exploiting Contextual Information via Dynamic Memory Network for Event Detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1030–1035, Brussels, Belgium. Association for Computational Linguistics. 51

Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. 2016. Leveraging FrameNet to improve automatic event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 2134–2143, Berlin, Germany. Association for Computational Linguistics. 22, 24, 26, 33

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 1789–1798, Vancouver, Canada. Association for Computational Linguistics. 22, 24

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022b. Dynamic prefix-tuning for generative template-based event extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 5216–5228, Dublin, Ireland. Association for Computational Linguistics. 20, 22, 24

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018b. Jointly Multiple Events Extraction via Attention-based Graph Information Aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256, Brussels, Belgium. Association for Computational Linguistics. 14, 16, 22, 24

Yang Liu, Jinpeng Hu, Xiang Wan, and Tsung-Hui Chang. 2022c. Learn from relation information: Towards prototype representation rectification for few-shot relation extraction. In *Findings of the Association for Computational Linguistics : NAACL 2022*, pages 1822–1831, Seattle, United States. Association for Computational Linguistics. 76

Wei Lu and Dan Roth. 2012. Automatic event extraction with structured preference modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 835–844, Jeju Island, Korea. Association for Computational Linguistics. 36

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. 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 2795–2806, Online. Association for Computational Linguistics. 22, 24, 34

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 5755–5772, Dublin, Ireland. Association for Computational Linguistics. 33, 34

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. 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 3036–3046, Minneapolis, Minnesota. Association for Computational Linguistics. 16

Qing Lyu, Hongming Zhang, Elior Sulem, and Dan Roth. 2021. Zero-shot event extraction via transfer learning: Challenges and insights. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2 : Short Papers)*, pages 322–332, Online. Association for Computational Linguistics. 33, 34, 36, 37, 100

Mingyu Derek Ma, Xiaoxuan Wang, Po-Nien Kung, P. Jeffrey Brantingham, Nanyun Peng, and Wei Wang. 2023a. Star : Boosting low-resource event extraction by structure-to-text data generation with large language models. 39

Yubo Ma, Yixin Cao, YongChing Hong, and Aixin Sun. 2023b. Large language model is not a good few-shot information extractor, but a good reranker for hard samples ! *ArXiv*, abs/2303.08559. 30, 39

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022. Prompt for extraction? PAIE: Prompting argument interaction for event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 6759–6774, Dublin, Ireland. Association for Computational Linguistics. 19, 22, 32, 35, 100, 113

Yubo Ma, Zehao Wang, Yixin Cao, and Aixin Sun. 2023c. Few-shot event detection: An empirical study and a unified view. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 11211–11236, Toronto, Canada. Association for Computational Linguistics. 24, 25, 39, 125

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics. 35

David Milne and Ian H. Witten. 2008. Learning to link with wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, page 509–518, New York, NY, USA. Association for Computing Machinery. 12

Minh Van Nguyen, Viet Dac Lai, and Thien Huu Nguyen. 2021. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 27–38, Online. Association for Computational Linguistics. 22, 24

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022. Joint extraction of entities, relations, and events via modeling inter-instance and inter-label dependencies. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 4363–4374, Seattle, United States. Association for Computational Linguistics. 17, 22, 24

Thien Nguyen and Ralph Grishman. 2018a. Graph convolutional networks with argument-aware pooling for event detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1). 105

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics. 14, 16, 22, 24

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2 : Short Papers)*, pages 365–371, Beijing, China. Association for Computational Linguistics. 15, 17, 22, 24, 105

Thien Huu Nguyen and Ralph Grishman. 2016. Modeling skip-grams for event detection with convolutional neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 886–891, Austin, Texas. Association for Computational Linguistics. 15, 22, 24

Thien Huu Nguyen and Ralph Grishman. 2018b. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press. 16, 17, 24

Trung Minh Nguyen and Thien Huu Nguyen. 2018a. One for all: Neural joint modeling of entities and events. In *AAAI Conference on Artificial Intelligence*. 13

Trung Minh Nguyen and Thien Huu Nguyen. 2018b. One for all : Neural joint modeling of entities and events. In *AAAI Conference on Artificial Intelligence*. 22, 24

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. 170

Iftitahu Nimah, Meng Fang, Vlado Menkovski, and Mykola Pechenizkiy. 2021. ProtoInfoMax: Prototypical networks with mutual information maximization for out-of-domain detection. In *Findings of the Association for Computational Linguistics : EMNLP 2021*, pages 1606–1617, Punta Cana, Dominican Republic. Association for Computational Linguistics. 78, 79, 81, 87

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2016. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *Proceedings of the 2nd Workshop on Computing News Storylines (CNS 2016)*, pages 47–56, Austin, Texas. Association for Computational Linguistics. 12

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *arXiv*, 2203.02155. 30

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. *ArXiv*. 19, 22

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 392–402, Austin, Texas. Association for Computational Linguistics. 32, 34, 36

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. 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. 29

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. Association for Computational Linguistics. 29, 57

Hoifung Poon and Lucy Vanderwende. 2010. Joint inference for knowledge extraction from biomedical literature. In *Human Language Technologies : The 2010 Annual Confe-*

rence of the North American Chapter of the Association for Computational Linguistics, pages 813–821, Los Angeles, California. Association for Computational Linguistics. 13, 14

Nicholas Popovic and Michael Färber. 2022. Few-shot document-level relation extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 5733–5746, Seattle, United States. Association for Computational Linguistics. 45

Amir Pouran Ben Veyseh, Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2021. Unleash GPT-2 Power for Event Detection. 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 6271–6282, Online. Association for Computational Linguistics. 19, 29

Viraj Prabhu, Anitha Kannan, Murali Ravuri, Manish Chablani, David Sontag, and Xavier Amatriain. 2018. Prototypical clustering networks for dermatological disease diagnosis. 76

James Pustejovsky. 1991. The syntax of event structure. *Cognition*, 41(1) :47–81. 37

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training. 19, 29

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140) :1–67. 19, 38

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*. 43

Sachin Ravi and H. Larochelle. 2016. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*. 5

Nils Reimers and Iryna Gurevych. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics. 58

Sebastian Riedel, Hong-Woo Chun, Toshihisa Takagi, and Jun'ichi Tsujii. 2009. A Markov Logic approach to bio-molecular event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 41–49, Boulder, Colorado. Association for Computational Linguistics. 13, 14

Sebastian Riedel and Andrew McCallum. 2011. Fast and robust joint models for biomedical event extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1–12, Edinburgh, Scotland, UK. Association for Computational Linguistics. 13, 14

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA. Association for Computational Linguistics. 37, 115

Luana Ruiz, Fernando Gama, and Alejandro Ribeiro. 2019. Gated graph convolutional recurrent neural networks. *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5. 17

Oscar Sainz, Itziar Gonzalez-Dios, Oier Lopez de Lacalle, Bonan Min, and Eneko Agirre. 2022a. Textual entailment for event argument extraction: Zero- and few-shot with multi-source learning. In *Findings of the Association for Computational Linguistics : NAACL 2022*, pages 2439–2455, Seattle, United States. Association for Computational Linguistics. 33, 36, 37, 100, 103, 108, 113, 115

Oscar Sainz, Haoling Qiu, Oier Lopez de Lacalle, Eneko Agirre, and Bonan Min. 2022b. ZS4IE: A toolkit for zero-shot information extraction with simple verbalizations. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies : System Demonstrations*, pages 27–38, Hybrid : Seattle, Washington + Online. Association for Computational Linguistics. 113

Evan Sandhaus. 2008. The new york times annotated corpus. 36, 115

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilic, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan,

Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, and et al. 2022. BLOOM: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100. 127

T. Schaul and J. Schmidhuber. 2010. Metalearning. *Scholarpedia*, 5(6) :4650. Revision #91489. 27

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics : Main Volume*, pages 255–269, Online. Association for Computational Linguistics. 21, 32

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics. 33

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web*, pages 593–607, Cham. Springer International Publishing. 106

Mike Schuster and Kuldip Paliwal. 1997. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45 :2673 – 2681. 16, 166

Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. 2001. Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 13(7) :1443–1471. 79

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1). 14, 22, 24, 120

Shirong Shen, Tongtong Wu, Guilin Qi, Yuan-Fang Li, Gholamreza Haffari, and Sheng Bi. 2021. Adaptive Knowledge-Enhanced Bayesian Meta-Learning for Few-shot Event Detection. In *Findings of the Association for Computational Linguistics : ACL-IJCNLP 2021*, pages 2417–2429, Online. Association for Computational Linguistics. 42

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc. 5, 7, 31, 42, 45, 50, 171

David R. So, Chen Liang, and Quoc V. Le. 2019. The evolved transformer. 171

Zhiyi Song, Ann Bies, Stephanie Strassel, Tom Riese, Justin Mott, Joe Ellis, Jonathan Wright, Seth Kulick, Neville Ryant, and Xiaoyi Ma. 2015. From light to rich ERE: Annotation of entities, relations, and events. In *Proceedings of the The 3rd Workshop on EVENTS : Definition, Detection, Coreference, and Representation*, pages 89–98, Denver, Colorado. Association for Computational Linguistics. 12

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. 2018. Learning to compare : Relation network for few-shot learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1199–1208. 5, 31, 46, 170

Ming Tan, Yang Yu, Haoyu Wang, Dakuo Wang, Saloni Potdar, Shiyu Chang, and Mo Yu. 2019. Out-of-domain detection for low-resource text classification tasks. 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 3566–3572, Hong Kong, China. Association for Computational Linguistics. 78, 79, 81, 87

Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B. Tenenbaum, and Phillip Isola. 2020. Rethinking few-shot image classification: A good embedding is all you need? In *Computer Vision – ECCV 2020 : 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV*, page 266–282, Berlin, Heidelberg. Springer-Verlag. 46

Meihan Tong, Bin Xu, Shuai Wang, Yixin Cao, Lei Hou, Juanzi Li, and Jun Xie. 2020. Improving event detection via open-domain trigger knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5887–5897, Online. Association for Computational Linguistics. 28, 29

MeiHan Tong, Bin Xu, Shuai Wang, Meihuan Han, Yixin Cao, Jiangqi Zhu, Siyu Chen, Lei Hou, and Juanzi Li. 2022. DocEE: A large-scale and fine-grained benchmark for document-level event extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 3970–3982, Seattle, United States. Association for Computational Linguistics. 163

Aboubacar Tuo, Romaric Besançon, Olivier Ferret, and Julien Tourille. 2022a. Mieux utiliser BERT pour la détection d'évènements à partir de peu d'exemples (better exploitation of BERT for few-shot event detection). In *Actes de la 29e Conférence sur le Traitement Automatique des Langues Naturelles. Volume 1 : conférence principale*, pages 392–402, Avignon, France. ATALA. 7, 68

Aboubacar Tuo, Romaric Besançon, Olivier Ferret, and Julien Tourille. 2023a. Détection d'événements à partir de peu d'exemples par seuillage dynamique. In *18e Conférence en Recherche d'Information et Applications – 16e Rencontres Jeunes Chercheurs en RI – 30e Conférence sur le Traitement Automatique des Langues Naturelles – 25e Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues*, pages 159–168, Paris, France. ATALA. 7

Aboubacar Tuo, Romaric Besançon, Olivier Ferret, and Julien Tourille. 2022b. Better Exploiting BERT for Few-Shot Event Detection. In *NLDB*, page 291–298, Berlin, Heidelberg. Springer-Verlag. 7, 28, 31, 68, 71, 72, 80, 91, 92

Aboubacar Tuo, Romaric Besançon, Olivier Ferret, and Julien Tourille. 2023b. Trigger or not trigger: Dynamic thresholding for few shot event detection. In *45th European Conference on Information Retrieval (ECIR 2023) : Advances in Information Retrieval, short article session*, volume 13981 of *Lecture Notes in Computer Science*, pages 637–645, Dublin, Ireland. Springer Nature Switzerland. 7, 28, 75

Marco A. Valenzuela-Escárcega, Gus Hahn-Powell, and Mihai Surdeanu. 2016. Odin's runes: A rule language for information extraction. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 322–329, Portorož, Slovenia. European Language Resources Association (ELRA). 4

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc. 14, 16, 49

Deepak Venugopal, Chen Chen, Vibhav Gogate, and Vincent Ng. 2014. Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 831–843, Doha, Qatar. Association for Computational Linguistics. 13, 14

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc. 5, 31, 43, 44, 45, 46, 72, 86, 100, 171

Ivan Vulić, Edoardo Maria Ponti, Anna Korhonen, and Goran Glavaš. 2021. LexFit: Lexical fine-tuning of pretrained language models. 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 5269–5283, Online. Association for Computational Linguistics. 49, 58, 60, 68

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. 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 5784–5789, Hong Kong, China. Association for Computational Linguistics. 16, 22, 24

Christopher Walker, Stephanie Strassel, and Kazuaki Maeda Julie Medero. 2006. *ACE 2005 Multilingual Training Corpus*. LDC corpora. Linguistic Data Consortium. 10, 85, 109

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B : A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax. 37

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015. A transition-based algorithm for AMR parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 366–375, Denver, Colorado. Association for Computational Linguistics. 36

Hanrui Wang, Zhanghao Wu, Zhijian Liu, Han Cai, Ligeng Zhu, Chuang Gan, and Song Han. 2020a. HAT: Hardware-aware transformers for efficient natural language processing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7675–7688, Online. Association for Computational Linguistics. 171

Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. 2019a. Panet: Few-shot image semantic segmentation with prototype alignment. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9196–9205. 5

Peiyi Wang, Runxin Xun, Tianyu Liu, Damai Dai, Baobao Chang, and Zhifang Sui. 2021a. Behind the scenes: An exploration of trigger biases problem in few-shot event classification. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 1969–1978, New York, NY, USA. Association for Computing Machinery. 47, 49, 64, 92

Sijia Wang, Mo Yu, and Lifu Huang. 2023a. The art of prompting: Event detection based on type specific prompts. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, pages 1286–1299, Toronto, Canada. Association for Computational Linguistics. 33

Xiaozhi Wang, Yulin Chen, Ning Ding, Hao Peng, Zimu Wang, Yankai Lin, Xu Han, Lei Hou, Juanzi Li, Zhiyuan Liu, Peng Li, and Jie Zhou. 2022. MAVEN-ERE: A unified large-scale dataset for event coreference, temporal, causal, and subevent relation extraction. In

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 926–941, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 12

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020b. MAVEN: A Massive General Domain Event Detection Dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1652–1671, Online. Association for Computational Linguistics. 12, 85, 161, 162, 163

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. 2019b. HMEAE: Hierarchical modular event argument extraction. 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 5777–5783, Hong Kong, China. Association for Computational Linguistics. 22, 24

Xingyao Wang, Sha Li, and Heng Ji. 2023b. Code4struct : Code generation for few-shot event structure prediction. 37

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021b. Automated concatenation of embeddings for structured prediction. 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 2643–2660, Online. Association for Computational Linguistics. 128

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juanzi Li, and Jie Zhou. 2021c. CLEVE: Contrastive Pre-training for Event Extraction. 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 6283–6297, Online. Association for Computational Linguistics. 22, 24

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv*, abs/2201.11903. 128

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, Yong Jiang, and Wenjuan Han. 2023. Zero-shot information extraction via chatting with chatgpt. 30, 128

Jin Xu, Xu Tan, Renqian Luo, Kaitao Song, Jian Li, Tao Qin, and Tie-Yan Liu. 2021. Nas-bert: Task-agnostic and adaptive-size bert compression with neural architecture search. In

Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, KDD '21, page 1933–1943, New York, NY, USA. Association for Computing Machinery. 171

Yinchuan Xu and Junlin Yang. 2019. Look again at the syntax: Relational graph convolutional network for gendered ambiguous pronoun resolution. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 96–101, Florence, Italy. Association for Computational Linguistics. 106

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. 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 5766–5770, Hong Kong, China. Association for Computational Linguistics. 17

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 289–299, San Diego, California. Association for Computational Linguistics. 13, 22, 24

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5284–5294, Florence, Italy. Association for Computational Linguistics. 19, 22, 24, 26

Xianjun Yang, Yujie Lu, and Linda Petzold. 2023a. Few-shot document-level event argument extraction. 28, 37

Xianjun Yang, Yujie Lu, and Linda Petzold. 2023b. Few-shot document-level event argument extraction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 8029–8046, Toronto, Canada. Association for Computational Linguistics. 45

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6365–6375, Online. Association for Computational Linguistics. 72, 86, 125

Deming Ye, Yankai Lin, Peng Li, and Maosong Sun. 2022. Packed levitated marker for entity and relation extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 4904–4917, Dublin, Ireland. Association for Computational Linguistics. 128

Pengfei Yu, Zixuan Zhang, Clare Voss, Jonathan May, and Heng Ji. 2022. Building an event extractor with only a few examples. In *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 102–109, Hybrid. Association for Computational Linguistics. 31, 34, 36, 42, 57, 100, 103

Zhenrui Yue, Huimin Zeng, Mengfei Lan, Heng Ji, and Dong Wang. 2023. Zero- and few-shot event detection via prompt-based meta learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 7928–7943, Toronto, Canada. Association for Computational Linguistics. 33

Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. In *Advances in Neural Information Processing Systems*, pages 11960–11970. 17

Ying Zeng, Yansong Feng, Rong Ma, Zheng Wang, Rui Yan, Chongde Shi, and Dongyan Zhao. 2018. Scale up event extraction learning via automatic training data generation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press. 26

Haoxing Zhang, Xiaofeng Zhang, Haibo Huang, and Lei Yu. 2022a. Prompt-based meta-learning for few-shot text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1342–1357, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 5

Hongming Zhang, Haoyu Wang, and Dan Roth. 2021a. Zero-shot Label-aware Event Trigger and Argument Classification. In *Findings of the Association for Computational Linguistics : ACL-IJCNLP 2021*, pages 1331–1340, Online. Association for Computational Linguistics. 34, 36, 100, 114, 115

Hongming Zhang, Haoyu Wang, and Dan Roth. 2021b. Zero-shot Label-Aware Event Trigger and Argument Classification. In *Findings of the Association for Computational Linguistics : ACL-IJCNLP 2021*, pages 1331–1340, Online. Association for Computational Linguistics. 42

Hongming Zhang, Wenlin Yao, and Dong Yu. 2022b. Efficient zero-shot event extraction with context-definition alignment. In *Findings of the Association for Computational Linguistics : EMNLP 2022*, pages 7169–7179, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 31, 34

Junchi Zhang, Yanxia Qin, Yue Zhang, Mengchi Liu, and Donghong Ji. 2019a. Extracting entities and events as a single task using a transition-based neural model. In *Proceedings*

of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, pages 5422–5428. International Joint Conferences on Artificial Intelligence Organization. 13, 22, 24

Ruihan Zhang, Wei Wei, Xian-Ling Mao, Rui Fang, and Dangyang Chen. 2022c. HCL-TAT: A hybrid contrastive learning method for few-shot event detection with task-adaptive threshold. In *Findings of the Association for Computational Linguistics : EMNLP 2022*, pages 1808–1819, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 31, 34, 71, 91, 92, 93

Senhui Zhang, Tao Ji, Wendi Ji, and Xiaoling Wang. 2022d. Zero-shot event detection based on ordered contrastive learning and prompt-based prediction. In *Findings of the Association for Computational Linguistics : NAACL 2022*, pages 2572–2580, Seattle, United States. Association for Computational Linguistics. 33

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022e. Opt : Open pre-trained transformer language models. 127

Tongtao Zhang, Heng Ji, and Avirup Sil. 2019b. Joint Entity and Event Extraction with Generative Adversarial Imitation Learning. *Data Intelligence*, 1(2) :99–120. 22, 24

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019c. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics. 103, 107

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022f. Transfer learning from semantic role labeling to event argument extraction with template-based slot querying. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2627–2647, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 36, 100

Zixuan Zhang and Heng Ji. 2021. Abstract Meaning Representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 39–49, Online. Association for Computational Linguistics. 22, 24

Kailin Zhao, Xiaolong Jin, Long Bai, Jiafeng Guo, and Xueqi Cheng. 2022. Knowledge-enhanced self-supervised prototypical network for few-shot event detection. In *Findings of the Association for Computational Linguistics : EMNLP 2022*, pages 6266–6275, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 31, 34

Kang Zhao, Hua Xu, Yue Cheng, Xiaoteng Li, and Kai Gao. 2021. Representation iterative fusion based on heterogeneous graph neural network for joint entity and relation extraction. *Knowledge-Based Systems*, 219 :106888. 128

Yue Zhao, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2018. Document embedding enhanced event detection with hierarchical and supervised attention. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers)*, pages 414–419, Melbourne, Australia. Association for Computational Linguistics. 22, 24

Yang Zhou, Yubo Chen, Jun Zhao, Yin Wu, Jiexin Xu, and Jinlong Li. 2021. What the role is vs. what plays the role: Semi-supervised event argument extraction via dual question answering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16) :14638–14646. 22, 24, 35, 100

Xinyu Zuo, Haijin Liang, Ning Jing, Shuang Zeng, Zhou Fang, and Yu Luo. 2022. Type-enriched hierarchical contrastive strategy for fine-grained entity typing. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2405–2417, Gyeongju, Republic of Korea. International Committee on Computational Linguistics. 128

Annexe A

Jeux de données

Nous donnons ici plus de détail sur les jeux de données ACE-2005, MAVEN et FewEvent que nous avons utilisés pour les expérimentations de cette thèse.

A.1 . ACE-2005

ACE-2005 est un jeu de données de référence pour l'extraction d'information, composé de 599 documents [1] de diverses sources, en anglais, mandarin et arabe. Il propose huit types d'événements, subdivisés en 33 sous-types et 34 rôles pour les arguments. Sa richesse et sa diversité en font une ressource clé pour le développement de modèles d'extraction d'information multilingues et polyvalents. Nous listons l'ensemble des types et sous-types dans le Tableau A.1.

Pour nos expérimentations, nous adoptons le découpage de Lai and Nguyen (2019), qui considère quatre types (Conflict, Business, Contact et Justice) dans l'ensemble d'entraînement et les autres types dans l'ensemble d'évaluation. Dans ce découpage, certains rôles n'apparaissent que dans l'ensemble d'évaluation tandis que d'autres sont exclusivement présents dans l'ensemble d'entraînement, mais il existe également des rôles communs aux deux ensembles (voir Tableau A.2). Nous avons supprimé les rôles d'argument comportant moins de 10 exemples afin de disposer de suffisamment de données pour constituer des épisodes avec au moins 10 exemples dans les ensembles de support. Nous avons également retiré les types pour lesquels il ne restait aucun argument (c'est-à-dire *Business:End-Org*, *Justice:Pardon*, *Justice:Extradite* et *Justice:Acquit* dans l'ensemble d'entraînement, ainsi que *Personnel:Nominate* dans l'ensemble d'évaluation).

A.2 . MAVEN

1. dans la version anglaise

Types	Sous-types	Arguments
Life	Be-born	Person, Place, Time
	Marry	Person, Place, Time
	Divorce	Person, Place, Time
	Injure	Agent, Victim, Instrument, Place, Time
	Die	Agent, Victim, Instrument, Place, Time
Movement	Transport	Agent, Artifact, Vehicle, Price, Origin, Destination, Time
Transaction	Transfer-Ownership	Buyer, Seller, Beneficiary, Artifact, Price, Place, Time
	Transfer-Money	Giver, Recipient, Beneficiary, Money, Place, Time
Business	Start-Org	Agent, Org, Place, Time
	Merge-Org	Org, Place, Time
	Declare-Bankruptcy	Org, Place, Time
	End-Org	Org, Place, Time
Conflict	Attack	Attacker, Target, Instrument, Place, Time
	Demonstrate	Entity, Place, Time
Contact	Meet	Entity, Place, Time
	Phone-Write	Entity, Time
Personnel	Start-Position	Person, Entity, Position, Place, Time
	End-Position	Person, Entity, Position, Place, Time
	Nominate	Person, Agent, Position, Place, Time
	Elect	Person, Entity, Position, Place, Time
Justice	Arrest-Jail	Person, Agent, Crime, Place, Time
	Release-Parole	Person, Entity, Crime, Place, Time
	Trial-Hearing	Defendant, Prosecutor, Adjudicator, Crime, Place, Time
	Charge-Indict	Defendant, Prosecutor, Adjudicator, Crime, Place, Time, Sentence
	Sue	Plaintiff, Defendant, Adjudicator, Crime, Place, Time
	Convict	Defendant, Adjudicator, Crime, Place, Time
	Sentence	Defendant, Adjudicator, Crime, Sentence, Place, Time
	Fine	Entity, Adjudicator, Money, Crime, Place, Time
	Execute	Person, Agent, Crime, Place, Time
	Extradite	Agent, Person, Destination, Origin, Crime, Time
	Acquit	Defendant, Adjudicator, Crime, Place, Time
	Appeal	Defendant, Prosecutor, Adjudicator, Crime, Place, Time
	Pardon	Defendant, Adjudicator, Crime, Place, Time

Table A.1 – Liste des types d'événements du jeu de données ACE-2005.

Ensemble	Arguments
Entraînement	Org, Adjudicator, Prosecutor, Defendant, Sentence, Plaintiff, Attacker, Target, Crime
Évaluation	Buyer, Seller, Recipient, Artifact, Vehicle, Position, Victim
Partagés	Agent, Person, Time, Place, Giver, Money, Entity, Instrument, Destination, Origin

Table A.2 – Répartition des rôles d'arguments entre l'ensemble d'entraînement et l'ensemble d'évaluation du jeu de données ACE-2005.

Le jeu de données MAVEN (*Massive General Domain Event Detection Dataset*), publié dans l'article de Wang et al. (2020b), est une ressource spécialement conçue pour la détection d'événements. À la différence d'ACE-2005, qui annote les entités, MAVEN se concentre exclusivement sur l'annotation des déclencheurs d'événements. Cette approche permet d'augmenter considérablement le nombre de types d'événements, passant ainsi de 33 types à 168 types annotés de façon automatique. MAVEN offre ainsi une base de données riche et diversifiée pour l'entraînement et l'évaluation de modèles de détection d'événements avec un large éventail de domaines. Toutefois, le fait qu'il soit annoté automatiquement conduit parfois à des erreurs d'annotation. Nous donnons l'ensemble des types d'événements de MAVEN à la Figure A.1.

Nous avons adopté le découpage proposé par Chen et al. (2021b) pour ce jeu de données. Ce découpage garantit également une séparation entre les types de l'ensemble d'entraînement et les types de l'ensemble d'évaluation.

A.3 . FewEvent

FewEvent (Deng et al., 2020) est un jeu de données spécialement conçu pour la détection d'événements à partir de peu d'exemples. Il comprend un total de 70 852 instances réparties en 19 types d'événements subdivisé en 100 sous-types. Chaque type d'événement est annoté avec environ 700 instances en moyenne. La construction de FewEvent s'est déroulée en deux phases : l'extension du nombre de types d'événements à partir de corpus existants tels que ACE-2005 et TAC-KBP-2017 [2] des campagnes TAC (*Text Analysis Conference*). Ensuite, l'introduction de nouveaux types d'événements, obtenus grâce à l'annotation automatique à l'aide de l'outil développé par Chen et al. (2017) [3]. Ces nouvelles catégories d'événements ont été extraites de sources telles que Freebase (Bollacker et al., 2008) et Wikipédia, tout en respectant des domaines thématiques spécifiques prédéfinis.

Nous résumons les statistiques sur ces jeux de données dans le Tableau A.3 ci-dessous.

2. TAC-KBP-2007
3. L'outil d'annotation *event-data*

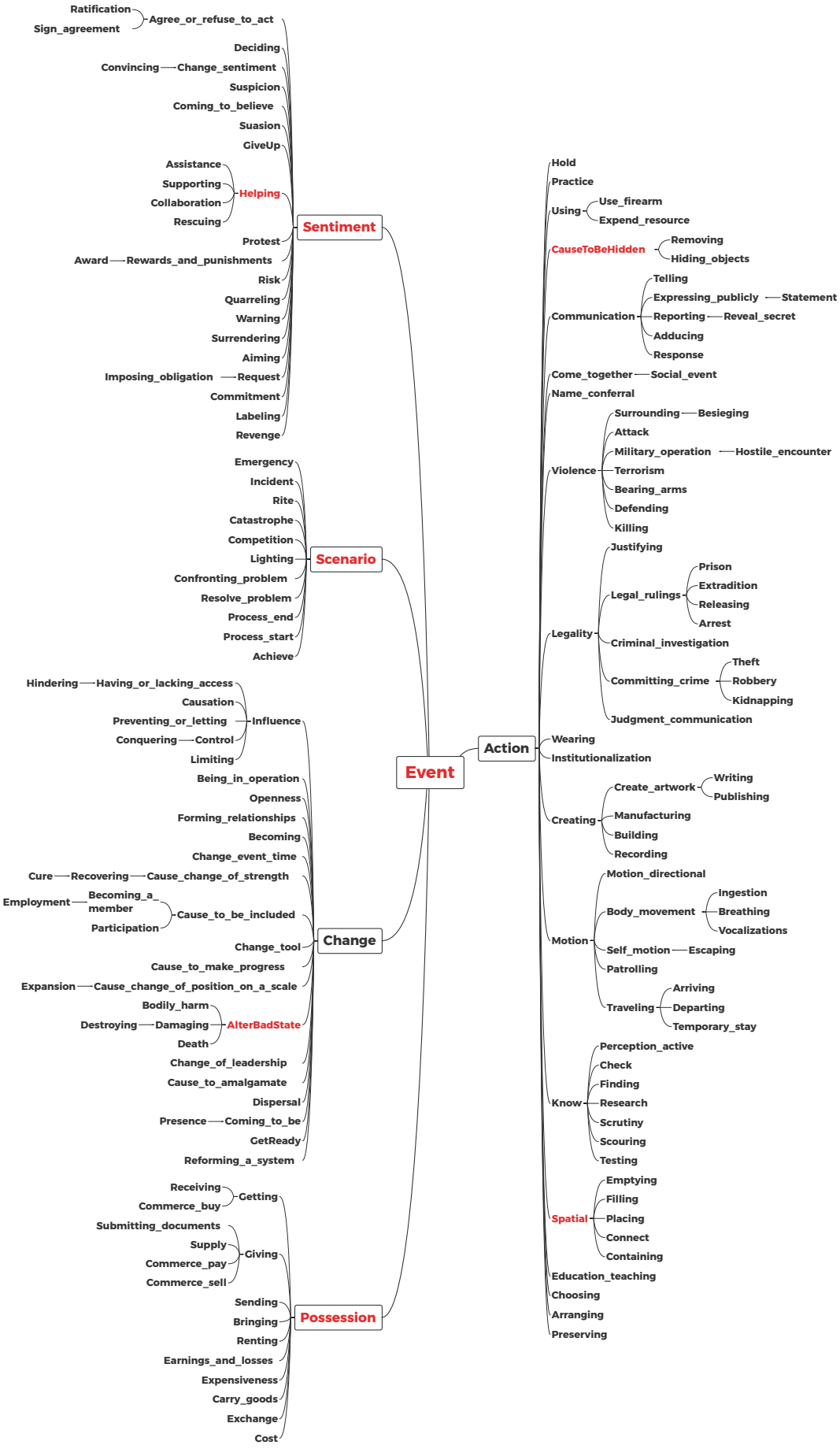


Figure A.1 – Les types d'événements du jeu de données MAVEN. Source : (Wang et al., 2020b).

		#Doc.	#Mots	#Phrases	#EvTyp.	#Evts	#EvInst.
ACE-2005 [4]		599	303k	15 789	33	4 090	5 349
Rich-ERE	LDC2015E29	91	43k	1 903	38	1 439	2 196
	LDC2015E68	197	164k	8 711	37	2 650	3 567
	LDC2015E78	171	114k	4 979	31	2 285	2 933
	TAC KBP 2014	351	282k	14 852	34	10 719	10 719
	TAC KBP 2015	360	238k	11 535	38	7 460	12 976
	TAC KBP 2016	169	109k	5 295	18	3 191	4 155
	TAC KBP 2017	167	99k	4 839	18	2 963	4 375
	Total	1 272	854k	41 708	38	29 293	38 853
MAVEN		4 480	1 276k	49 873	168	111 611	118 732
FewEvent†		-	-	70 852	100	70 852	70 852

Table A.3 – Comparaison entre les jeux de données ACE-2005, Rich-ERE, MAVEN et FewEvent. Le nombre de mots est obtenu à partir du découpage fait par NLTK (Bird and Loper, 2004). Ces statistiques sont issues de Wang et al. (2020b) et Deng et al. (2020). †FewEvent est directement découpé en phrases et ne prend pas en compte la notion de coréférence d'événements. Les nombres de phrases, d'événements et de mentions sont donc égaux.

A.4 . Jeux de données au niveau document

Plusieurs autres travaux récents ont également proposé des jeux de données d'extraction d'arguments d'événements au niveau du document, comme dans les campagnes MUC. Par exemple, RAMS (Ebner et al., 2020) combine l'annotation des rôles sémantiques et la résolution de coréférence pour passer du niveau phrastique au niveau document, ce qui permet une analyse plus globale des arguments. WikiEvents (Li et al., 2021) collecte des articles de Wikipédia décrivant des événements du monde réel puis utilise les liens de référence pour récupérer des articles de presse associés. Enfin, DocEE (Tong et al., 2022) vise à étendre l'ontologie des corpus existants à plus grande échelle, permettant par exemple de passer de 9 000 documents dans RAMS à plus de 27 000. Nous donnons plus de détails sur ces corpus dans le Tableau A.4.

Ces jeux de données ont été développés dans le but d'étendre l'extraction d'événements au-delà du niveau phrastique et de capturer les relations et les arguments d'événements à l'échelle du document. En effet, dans les cas d'usage réels, les arguments d'un même événement ne sont pas nécessairement regroupés au sein d'une seule et même phrase. Par exemple, dans un article de presse, on retrouve en général le déclencheur dans le titre et les arguments dans le chapeau et le corps de l'article.

Toutefois, dans cette étude, nous nous concentrons exclusivement sur l'extraction d'événements au niveau phrastique. Nous renvoyons à la revue de Liu et al. (2023b) qui est, à notre connaissance, la revue la plus récente sur la problématique de l'extraction au niveau document.

Corpus	#Doc.	#EvTyp.	#ArgTyp.	#Phrases	#ArgInst.	Portée
ACE-2005*	599	33	34	15 789	9 590	1
MUC-4	1 700	4	5	21 928	2 641	4,0
RAMS	9 124	139	65	34 536	21 237	4,8
WikiEvents	246	59	50	8 544	5 536	2,2
DocEE	27 485	59	356	749 568	180 528	10,2

Table A.4 – Statistiques sur les jeux de données d'extraction d'événements à l'échelle du document. ACE-2005* est annoté au niveau phrastique, mais sert de point de comparaison. (Doc : document, EvTyp. : types d'événement, ArgTyp. : types d'arguments, ArgInst. : les instances d'arguments, Portée : le nombre de phrases en moyenne dans lesquelles les arguments d'événements du même événement sont dispersés.

Annexe B

Modèles neuronaux pour l'extraction d'événements

Dans cette annexe, nous présentons plus en détail les principales architectures neuronales servant d'extracteurs de descripteurs pour la tâche d'extraction d'événements avant l'avènement des *transformer* et des *LLM*. Nous présentons notamment les réseaux de neurones convolutifs, les réseaux de neurones récurrents et les réseaux convolutifs sur des graphes.

B.1 . Réseaux de neurones convolutifs (CNN)

L'une des architectures neuronales les plus utilisées est le réseau neuronal convolutif proposé par LeCun et al. (1989) en 1989 pour la reconnaissance de caractères manuscrits. Elle est constituée de plusieurs couches de convolution permettant l'extraction automatique de descripteurs syntaxiques et sémantiques au sein des phrases (voir Figure B.1). Une couche de convolution est constituée d'un filtre de taille k couvrant k mots consécutifs. Dans une tâche de classification de mots, comme pour l'extraction d'événements, chaque mot est représenté dans un contexte local, de taille fixe m, centré sur le mot en question. Soit u le vecteur d'un filtre de taille k, la sortie de la couche de convolution l pour le mot i est donnée par :

$$h_i^l = \sigma\left(\sum_{a=0}^{k-1} u_a h_{i+a}^l\right) \tag{B.1}$$

où $\sigma(.)$ est une fonction d'activation.

B.2 . Réseaux de neurones récurrents (RNN)

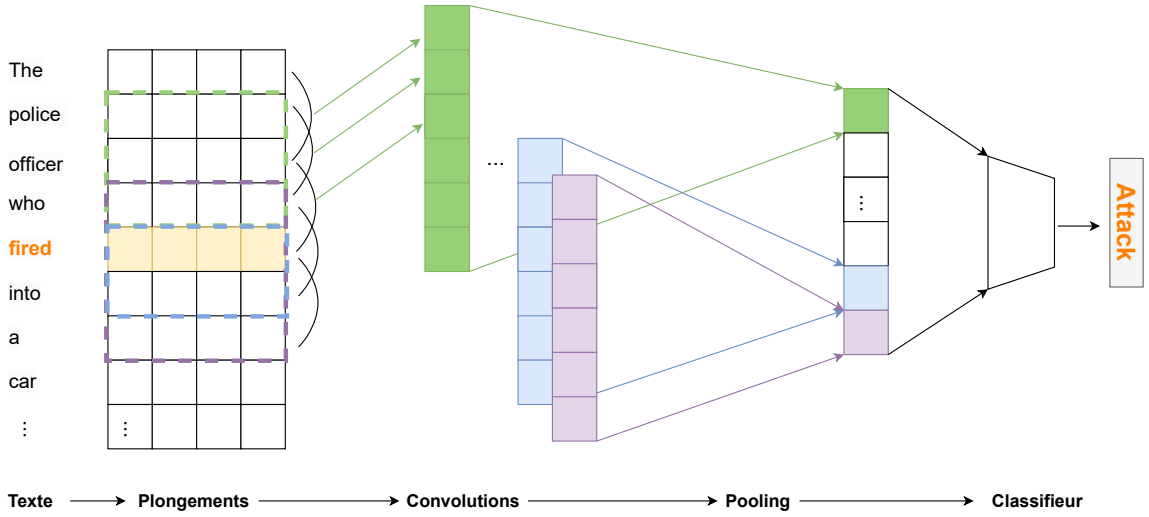


Figure B.1 – Réseau de neurones convolutif pour la classification du mot ***fired***.

À l'inverse des réseaux convolutifs, les réseaux de neurones récurrents sont plus adaptés à la modélisation des interactions plus longues au sein des passages et moins sensibles à la position de mots. L'idée principale des réseaux récurrents est de construire de manière récursive les représentations de chaque mot en prenant compte la représentation du mot précédent. Pour une séquence $\mathbf{x} = (x_1, \ldots, x_L)$ de longueur L, la représentation du mot x_i est donnée par :

$$h_i = \text{RNN}(\mathbf{x}_{1:i}) = \sigma\left(\mathbf{W_i x_i} + \mathbf{W_{i-1} x_{i-1}}\right) \tag{B.2}$$

où $\sigma(.)$ est une fonction d'activation et $\mathbf{W_i}$, les poids appris.

Une telle architecture présente deux inconvénients majeurs : d'une part, elle ne considère que le contexte de gauche, et d'autre part, elle est soumise au problème dit de la disparition du gradient (*gradient vanishing*) qui arrive lorsque les gradients utilisés pour mettre à jour les poids du réseau diminuent progressivement à mesure que l'on s'éloigne des premiers éléments d'une séquence. Le *gradient vanishing* se produit principalement en raison de l'utilisation de fonctions d'activation non linéaires (telles que la fonction sigmoïde). Lorsque les gradients sont multipliés par ces petites valeurs à chaque rétro-propagation, ils tendent à s'approcher de zéro, réduisant ainsi l'impact des représentations plus éloignées de la sortie. Pour résoudre ce problème, Hochreiter and Schmidhuber (1997) et Cho et al. (2014) proposent respectivement les LSTMs (*Long Short Time Memory)* et les GRUs (*Gated Recurrent Unit*), qui utilisent un mécanisme de porte pour introduire une notion de « mémoire ». Afin de capturer à la fois le contexte avant et après de chaque mot considéré, les réseaux bidirectionnels ont été introduits par Schuster and Paliwal (1997) en 1997. Ces réseaux lisent la séquence dans les deux sens (avant, *forward*) et arrière, *backward*) et concatènent les représentations obtenues pour chaque mot, permettant ainsi de capturer des informations contextuelles plus riches.

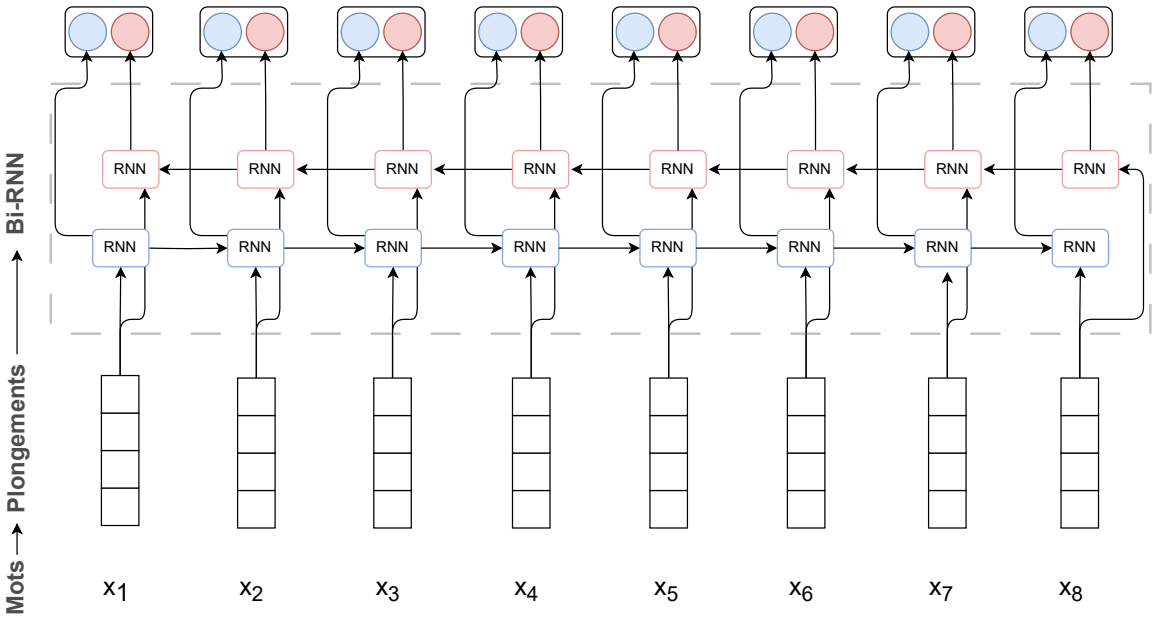


Figure B.2 – Architecture d'un réseau de neurones récurrent bidirectionnel.

B.3 . Réseaux de convolution sur des graphes

Les modèles présentés précédemment cherchent surtout à extraire des descripteurs au niveau phrastique ou du document en exploitant les informations contextuelles dans les séquences de mots. Toutefois, les événements peuvent être représentés sous forme de graphe dont les nœuds correspondent aux déclencheurs et aux arguments tandis que les arrêtes y représentent les rôles de ces arguments. Par ailleurs, une phrase peut également être représentée selon une structure de graphe, par exemple, à travers une analyse en dépendances syntaxiques ou sémantiques. Ces structures ont inspiré des travaux pour l'extraction d'événements avec l'avènement des réseaux de convolution sur les graphes (*Graph Convolutionnal Networks*, GCN) en 2017 (Kipf and Welling, 2017a) pour la classification de nœuds dans un graphe. Les GCN sont assez similaires aux réseaux de neurones convolutifs mais s'appliquent directement sur des graphes. Ils permettent notamment de capturer des dépendances plus lointaines au sein des phrases au lieu de se limiter aux mots directement voisins dans la séquence. Cela peut être particulièrement efficace pour l'extraction d'événements. En effet, cela permet de porter plus d'attention sur les relations sémantiques et syntaxiques entre les déclencheurs et les arguments plutôt que de parcourir toute la séquence. De la même manière que dans les réseaux de convolution standard (Lecun et al., 1998), en empilant des couches de GCN, on peut incorporer un voisinage plus lointain dans le graphe.

Formellement, soit un graphe $\mathcal{G}(\mathcal{V}, \mathcal{E})$ constitué des nœuds $\mathcal{V}$ et des arêtes $\mathcal{E}$. Dans le cadre de l'extraction d'événements, l'ensemble des nœuds représente les mots et les arêtes sont les relations entre eux. Chaque nœud i est représenté par un plongement contextuel h_i^l sur la couche de convolution l.

Le plongement d'un nœud à la sortie de la couche $l+1$ est donné par :

$$h_i^{l+1} = \sigma\left(\sum_{(i,j)\in\mathcal{E}} \alpha_{ij} W^l h_j^l + b^l\right)$$

où α_{ij} est un poids associé à l'arête (i,j), W^l et b^l sont des poids appris. Sous sa forme matricielle, cette expression s'écrit :

$$H^{l+1} = \sigma\left(\alpha^l W^l H^l A + b^l\right)$$

avec A, la matrice d'adjacence du graphe et H^l, la concaténation de tous les h_i^l. C'est souvent la forme matricielle qui est utilisée dans les implémentations.

Annexe C

Le méta-apprentissage

De manière générale, l'apprentissage profond vise à optimiser les paramètres θ d'un modèle f_θ en minimisant la fonction de coût $\mathcal{L}(\theta, \mathcal{D})$ sur l'ensemble de données d'entraînement $\mathcal{D}$. Cette fonction de coût est ensuite minimisée par un algorithme d'optimisation (par exemple, la descente de gradient) pour obtenir les paramètres optimaux $\theta^* = \arg\min_\theta \mathcal{L}(\theta, \mathcal{D})$.

En méta-apprentissage, on cherche plutôt à apprendre un algorithme d'apprentissage. Nous pouvons voir cela comme l'apprentissage d'une fonction $F_\phi(.)$ paramétrée par les paramètres ϕ dont l'objectif est de trouver les paramètres optimaux θ^*. Cette approche est souvent décrite comme le fait d'apprendre à apprendre. En effet, nous cherchons à apprendre les paramètres θ via l'apprentissage des méta-paramètres ϕ. Les méta-paramètres ϕ peuvent par exemple représenter l'architecture du modèle, l'initialisation des paramètres θ, le taux d'apprentissage, l'algorithme d'optimisation, etc.

L'apprentissage de ces méta-paramètres ϕ se fait via des méta-tâches d'entraînement :

$$\mathcal{T}_{train} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_N\}$$

où les tâches $\mathcal{T}_i$ peuvent être issues du même problème ou non.

Chaque tâche $\mathcal{T}_i$ est définie par un ensemble de support (*support set*) $\mathcal{S}_i$ et un ensemble query (*query set*) $\mathcal{Q}_i$ constitués de données $\mathbf{x} = \{x_1, x_2, \ldots, x_K\}$ alignées sur leurs annotations $\{y_1, y_2, \ldots, y_K\}$ comme dans un problème d'apprentissage standard.

Une tâche $\mathcal{T}_i$ peut être considérée comme un problème d'apprentissage standard ayant pour paramètres θ^i où le support set joue le rôle d'ensemble d'apprentissage, tandis que le query set peut être vu comme l'ensemble de test. L'on peut donc définir une fonction de coût sur les paramètres ϕ :

$$L(\phi, \mathcal{T}_{train}) = g\left(\mathcal{L}(\theta^i, \mathcal{T}_i)\right)$$

où $g(.)$ est une fonction d'agrégation.

Les méta-paramètres pouvant être appris comme pour un problème d'apprentissage standard ne parcourant les tâche $\mathcal{T}_i$ de $\mathcal{T}_{train}$. Une itération de mise à jour des méta-paramètres ϕ est appelée un épisode (*episode*).

Ce nouveau paradigme d'apprentissage permet d'améliorer la capacité d'adaptation d'un modèle à de nouvelles tâches et ainsi de réduire les efforts d'annotations spécifiques à chaque tâche.

Les différentes approches de méta-apprentissage peuvent être classifiées en trois grands groupes selon les objectifs visés. Les méthodes par comparaison se concentrent sur l'apprentissage de la représentation latente des entrées et des métriques de distance pour la comparaison, les méthodes par optimisation cherchent à apprendre comment initialiser de façon efficace les hyper-paramètres d'un réseau donné et la recherche d'architecture de réseau (*Network Architecture Search*, NAS) se focalise sur l'apprentissage de l'architecture du réseau.

C.1 . Les approches par optimisation (*optimisation-based*)

Les approches par optimisation visent à apprendre les hyper-paramètres des algorithmes d'optimisation. `MAML` (*Model-Agnostic Meta-Learning*) (Finn et al., 2017a) est le travail le plus représentatif dans ce contexte. Il vise à trouver des paramètres initiaux pouvant être rapidement adaptés à de nouvelles tâches en seulement quelques étapes d'entraînement. Dans ce contexte, le méta-paramètre ϕ correspond simplement aux valeurs initiales des paramètres θ qui serviront à initialiser le modèle sur de nouvelles tâches. Plusieurs autres travaux ont suivi, tels que `FOMAML` et `Reptile` (Nichol et al., 2018), qui cherchent à réduire la complexité de `MAML` pour la recherche des paramètres initiaux. La méthode `DG-MAML` (*Domain Generalization MAML*) (Li et al., 2018) propose un algorithme permettant l'adaptation au domaine tandis que `LEOPARD` et `Proto(FO)MAML` étendent le cadre de travail MAML à des architectures de modèles différentes.

C.2 . Les approches qui apprennent à comparer (*metric-based*)

Ces approches se concentrent sur la comparaison des exemples en utilisant leurs représentations latentes. Elles sont particulièrement efficaces dans les problèmes de classification, où elles exploitent les distances ou les similarités entre les exemples pour prendre des décisions de classification. L'objectif de ces approches est d'apprendre une métrique et des représentations latentes de sorte à regrouper les exemples de la même classe tout en écartant les classes les unes des autres. Le méta-paramètre appris est la métrique de comparaison, comme dans les réseaux de relation (*relation networks*) (Sung et al., 2018). L'un des avantages principaux de ces méthodes est qu'elles deviennent non paramétriques dès lors que la métrique de comparaison est fixée à l'avance,

comme dans le cas des réseaux prototypiques (*prototypical networks*)(Snell et al., 2017), les réseaux de correspondance (*matching networks*) (Vinyals et al., 2016)) ou encore des réseaux d'induction (*induction networks*) (Geng et al., 2019). Cette caractéristique les rend beaucoup plus rapides à entraîner par rapport à des approches telles que MAML. Nous donnons plus de détails sur ces méthodes dans le Chapitre 3 puisque les contributions de cette thèse reposent essentiellement sur ces méthodes.

C.3 . Les approches fondées sur l'architecture des modèles (*model-based* ou *Network Architecture Search*, NAS)

Les approches fondées sur l'architecture des modèles visent quant à elles à apprendre la meilleure architecture de modèle pour un problème donné. Elles cherchent à automatiser l'ingénierie d'architecture, qui est une étape clé dans le développement des modèles d'apprentissage profond aujourd'hui. Elles peuvent être vues comme un sous-domaine de l'apprentissage statistique automatique (*AutoML*)[1] (Elsken et al., 2019). Malgré les performances prometteuses de ces méthodes dans le domaine du TAL, qui ont donné lieu à des modèles tels que Evolved Transformer (So et al., 2019), HAT (Wang et al., 2020a), AdaBERT (Chen et al., 2021a), NAS-BERT (Xu et al., 2021) ou encore EfficientBERT (Dong et al., 2021), elles sont encore beaucoup moins répandues que les deux approches précédentes.

1. AutoML