Semantic-based approaches to enhance sentiment analysis quality

Wissam Mammar Kouadri

Université de Paris

LIPADE
Laboratoire d'Informatique PAris DEscartes

Semantic-Based Approaches to Enhance Sentiment Analysis Quality

Thèse de doctorat de Université de Paris

École doctorale n°130 EDITE
Spécialité de doctorat: Intelligence artificielle

Thèse présentée et soutenue à Paris, 24 November 2021, par

Wissam MAMMAR KOUADRI

Composition du jury :

Mme. Sihem Amer Yahia Directrice de Recherche, CNRS Grenoble	Rapporteur
M. Pierre Senellart Professeur, ENS Paris, Université PSL	Rapporteur
Mme. Amel Bouzeghoub Professeur, Institut Polytechnique de Paris	Examinatrice
M. Themis Palpanas Professeur, Université de Paris	Examinateur
Mme. Salima Benbernou Professeur, Université de Paris	Directrice de thèse
M. Mourad Ouziri MCU, Université de Paris	Invité
M. Iheb Ben Amor Docteur, IMBA Consulting	Invité

Thèse de doctorat

Contents

List of Figures

List of Tables

Abstract

The opinion expressed on various websites and social media is crucial to several organizations' decision-making processes. Sentiment analysis, a.k.a. opinion mining, is a process that involves automatically identifying the polarity of an opinion in the text (e.g., Positive/Neutral/Negative). However, despite the advance of research, sentiment analysis tools provide inconsistent polarities, harmful to business decisions.

In this PhD work, we focus on studying the inconsistencies of sentiment analysis tools as a data quality issue.

First, we carried out an in-depth empirical assessment of the quality of the sentiment analysis tool. Our evaluation involves two types of inconsistency: intra-tool inconsistency, which predicts different polarity for semantically equivalent documents by the same tool, and inter-tool inconsistency, which predicts different polarity for semantically equivalent documents across different tools.

Then, we introduce SAQ, a novel tool based on the Markov logic network that fixes both types of inconsistencies. Furthermore, through an empirical study, we investigate the impact of resolving inconsistency on tools' accuracy. Our results are promising and point to the improvement obtained by resolving both intra-tool and inter-tool inconsistency.

Further, we propose WSSA, a semantic-based weakly-supervised approach for sentiment analysis that aggregates labels from several weak sources. The aggregation process in WSSA is based on probabilistic soft logic reasoning that we enhance by a new learning procedure. In addition to resolving the two types of inconsistencies, it involves domain expert knowledge that guides the aggregation and preference order between documents established based on their subjectivity. The experimental evaluation has proved the efficiency of WSSA in polarity classification.

Key Words: Sentiment analysis, data quality, inconsistency resolution, weak supervision, and data integration.

Résumé

L'analyse de sentiments est le processus d'extraction de la polarité d'un texte. Malgré les avancées de la recherche réalisées dans ce domaine, cette tâche reste difficile à cause de la richesse du langage naturel et la dépendance de la polarité au contexte. En effet, nos expérimentations ont montré que les outils d'analyse de sentiment présentent des incohérences et manquent de qualité de prédiction. Nous avons observé sur des données réelles, que les différents algorithmes d'analyse de sentiment attribuent des différentes polarités au même texte (ex : tweet). Ainsi, deux tweets sémantiquement équivalents sont classifiés différemment par le même algorithme. Ceci est traduit par (1) des incohérences entre algorithmes : deux algorithmes donnent des sorties différentes pour la même entrée,(2)des incohérences intra algorithme: l'algorithme donne des résultats différents pour des entrées sémantiquement équivalentes.

Motivé par ces observations, les travaux de recherche dans le domaine de l'apprentissage automatique contradictoire (adversarial machine learning) et l'analyse de sentiment, le travail de thèse consiste à quantifier le phénomène des incohérences dans les outils d'analyse du sentiment, déterminer les causes et les facteurs responsables de ses incohérences, puis proposer une méthode qui résout ses incohérences et enfin étudier l'effet de résoudre les incohérences sur la précision. En se basant sur les résultats des études effectuées, nous proposons une méthode d'analyse de sentiment par supervision faible.

La première partie de la thèse est consacrée à étudier le phénomène d'incohérence intra et inter-algorithmes dans les outils d'analyses de sentiment via une étude empirique extensive sur plusieurs axes (statistique, structurel et sémantique), pour déterminer les causes et les facteurs qui vont influencer sur les incohérences, ainsi que la création d'un benchmark de test et la proposition d'une heuristique pour affiner sa qualité. Nos résultats ont montré que les incohérences sont fréquentes dans toutes les catégories d'algorithmes d'analyse de sentiment.

La deuxième partie est dédiée à la proposition d'une méthode pour résoudre les deux types d'incohérences intra et inter-algorithmes et d'étudier l'effet de la résolution de ces incohérences sur la précision. Pour cela, nous avons développé SAQ, une méthode basée sur la logique probabiliste de Markov (MLN) qui fusionne les etiquettes prevenant de différents algorithmes en résolvant les deux types d'incohérences et améliore la précision. Nous avons étudié l'efficacité de SAQ et nous l'avons comparé à des méthodes de résolution d'incohérence dans le crowdsourcing sur plusieurs datasets. Nos résultats montrent que résoudre les deux types d'incohérence améliore la précision.

Motivés par nos résultats et amélioration de la précision obtenue par la résolution des incohérences, dans la troisième partie de thèse, nous avons proposé WSSA, une méthode d'analyse de sentiment basée sur le paradigme de la classification faible (weak supervision), qui consiste à considérer plusieurs outils d'analyse de sentiment comme source d'étiquettes faibles, puis réduit les incohérences entre ces algorithmes en proposant un algorithme itératif qui permet de classer les algorithmes selon leurs pondérations et inférer la polarité optimale du système en utilisant des mécanismes d'inférence logique probabiliste sur la base du modèle Probabilistic Soft Logic.

Mots clés: Analyse des sentiments, qualité des données, résolution des incohérences, supervision faible et intégration des données.

Résumé

L'analyse de sentiments est le processus automatique qui permet d'extraire la polarité (positive, négative ou neutre) d'un texte. Malgré les avancées de la recherche réalisées dans ce domaine, cette tâche reste difficile à cause de la richesse du langage naturel et la dépendance de la polarité au contexte.

En effet, nos expérimentations ont montré que les outils d'analyse de sentiment présentent des incohérences et manquent de qualité de prédiction. Nous avons observé sur des données réelles, que les différents algorithmes d'analyse de sentiment attribuent des différentes polarités au même texte (ex : tweet). Ainsi, deux tweets sémantiquement équivalents sont classifiés différemment par le même algorithme. Ceci est traduit par (1) des incohérences entre algorithmes : deux algorithmes donnent des sorties différentes pour la même entrée, (2) des incohérences intra algorithme: l'algorithme donne des résultats différents pour des entrées sémantiquement équivalentes.

Motivé par ces observations, les travaux de recherche dans le domaine de l'apprentissage automatique contradictoire (adversarial machine learning) et l'analyse de sentiment, le travail de cette thèse consiste à quantifier le phénomène des incohérences dans les outils d'analyse du sentiment, déterminer les causes et les facteurs responsables de ses incohérences, proposer une méthode qui résout ses incohérences et enfin, étudier l'effet de résoudre les incohérences sur la précision. En se basant sur les résultats des études effectuées, nous proposons une méthode d'analyse de sentiment par supervision faible.

La première partie de la thèse est consacrée à étudier le phénomène d'incohérence intra et inter-algorithmes dans les outils d'analyses de sentiment via une étude empirique extensive sur plusieurs axes (statistique, structurel et sémantique), pour déterminer les causes et les facteurs qui vont influencer les incohérences. Pour résumer, nous avons proposé:

- **Des algorithmes génériques pour les outils d'analyse de sentiment**

  Nous avons effectué, dans un premier temps, une étude bibliographique sur les méthodes d'analyse de sentiment, proposé des abstractions algorithmiques pour chaque catégorie d'algorithmes et sélectionné six algorithmes représentatifs à évaluer.

- **Évaluation Empirique**

  Nous avons mené une étude empirique qui qualifie les incohérences et explique la non-robustesse des algorithmes d'analyse de sentiment en présence de paraphrases (des phrases sémantiquement équivalentes). Notre évaluation couvre deux types d'incohérences : des incohérences intra-algorithmes et des incohérences interalgorithmes, et considère les axes suivants:

  Étude Statistique. Cette étude (1) vérifie si les incohérences sont fréquentes ou s'il s'agit d'anomalies rares, (2) trouve les différents types d'incohérences (3) et détermine les algorithmes et le type de données qui sont plus vulnérables à l'incohérence. Les résultats ont montré que les incohérences sont très fréquentes sur toutes les catégories d'algorithmes et dans tous les ensembles de données avec plus de présence dans les algorithmes basés sur l'apprentissage profond et sur les données factuelles.

Étude structurelle. Dans cette étude nous nous intéressons à la relation entre les incohérences et la structure des paraphrases évaluées en focalisant sur l'impact des similarités sémantiques et syntaxiques sur les incohérences. Les résultats ont montré que les algorithmes sont très sensibles à la différence syntaxique entre les textes sémantiquement équivalents.

Étude sémantique. Le but de cette étude est de vérifier si les incohérences dépendent de la subjectivité des paraphrases et s' il y a une relation entre la précision de l'algorithme et les incohérences qu'il présente. Autrement dit, nous vérifions si les algorithmes les plus précis présentent moins d'incohérence. Les résultats ont montré plus d'incohérence entre les données factuelles ainsi qu'une corrélation inverse entre les incohérences inter-algorithmes et la précision, i.e, plus l'algorithme est cohérent, plus il est plus précis.

Étude de l'effet des hyperparamètres sur la précision et les incohérences. Étant donné le constat effectué sur les outils basés sur l'apprentissage profond, nous avons étudié l'impact des hyperparamètres sur la précision et l'incohérence des algorithmes. Les résultats ont montré que les incohérences sont présentes sur toutes les configurations. En se basant sur les résultats, nous avons suggéré des configurations qui permettent de minimiser les incohérences et maximiser la précision.

- **Proposition d'un benchmark de test**

  Afin d'évaluer les incohérences et la précision des algorithmes, nous avons construit un benchmark de paraphrases étiquetées avec des polarités. Le benchmark est construit à partir de cinq corpus pour l'analyse de sentiment, disponibles publiquement, que nous avons augmenté avec des paraphrases en utilisant la méthode [1]. La méthode que nous avons utilisée a une précision de 80%. Afin d'augmenter la qualité de notre benchmark, nous avons proposé une heuristique qui permet de réduire la marge d'erreur en gardant que les paraphrases valides. Cette heuristique permet de minimiser l'effort humain pour la vérification de la qualité des données et assure un benchmark de qualité avec un taux d'erreur réduit.

- **Recommandations**

  En se basant sur les résultats d'évaluation, nous proposons un ensemble de recommandation pour choisir les algorithmes d'analyse de sentiment adaptés à un scénario donné.

La deuxième partie est dédiée à la proposition d'une méthode pour résoudre les deux types d'incohérence intra et inter-algorithmes et d'étudier l'effet de résoudre les incohérences sur la précision:

- **Proposition d'une plate-forme pour la résolution des incohérences intra et inter-algorithmes dans les algorithmes d'analyse de sentiment**

  Plusieurs travaux de recherche ont proposé des méthodes pour résoudre les incohérences afin d'améliorer la précision des systèmes comme les travaux de [2], [3].

Cependant ces travaux ont étudié les deux types d'incohérence séparément et à notre connaissance, il n'y aucun travail qui a exploré la résolution des deux types d'incohérence simultanément pour concevoir des outils plus performants. D'où l'idée de SAQ une approche basée sur les réseaux logiques probabilistes de Markov qui résout les incohérences intra et inter-algorithmes dans les outils d'analyse de sentiment et améliore la qualité des systèmes.

L'intuition de SAQ est de résoudre les incohérences pour réduire le nombre de prédictions erronées. Ce qui permet la convergence vers les données de référence (golden truth).

- **Etude de l'impact de résoudre les incohérences sur la précision des algorithmes**

  Nous avons étudié l'effet de résoudre les incohérences sur la précision des algorithmes. Pour cela, nous avons comparé SAQ à plusieurs approches de l'état-de-l'art pour la résolution des incohérences. Puis nous avons étudié l'effet résoudre les incohérences en variant plusieurs facteurs tels que le nombre des algorithmes d 'analyse de sentiment à considérer dans la résolution des incohérences et la précision primaire de ces algorithmes.

  Les résultats de cette étude ont permis de déterminer des cas de figure qui ne garantissent pas l'amélioration de précision et montrer l'efficacité de SAQ.

  En effet, les résultats de SAQ sont promoteurs, montrent l'efficacité de considérer la similarité sémantique entre les documents dans la résolution des incohérences et pointe une amélioration de précision de 20%.

Dans la troisième partie de la thèse nous avons exploité nos résultats pour proposer une nouvelle approche pour l'analyse de sentiment basée sur le paradigme de supervision faible:

- **WSSA: Un algorithme sémantique faiblement supervisé pour l'analyse de sentiment**

  Motivés par nos résultats, nous avons proposé WSSA, une approche d'analyse de sentiment basée sur le paradigme de la supervision faible (weak supervision), qui considère plusieurs outils d'analyse de sentiment comme sources d'étiquettes faibles, puis réduit les incohérences entre ces algorithmes en fonction de plusieurs facteurs: l'incohérence entre les sources d'étiquettes faibles, la cohérence entre ces sources, la subjectivité des documents, et la similarité sémantique entre les documents. Pour guider le raisonnement, WSSA permet d'intégrer aussi les connaissances d'experts du domaine sous forme de règles.

  Pour assurer la scalabilité de WSSA, nous avons utilisé le framework logique probabiliste scalable (PSL) qui permet de modéliser les incohérences, exprimer l'incertitude ainsi que d'intégrer les règles du domaine.

- **Algorithme non supervisé pour l'apprentissage des pondérations de règles**

  Puisque l'apprentissage dans PSL est supervisé et nécessitant des polarités de référence (ground truth), nous avons proposé un nouvel algorithme itératif non

supervisé qui classifie les outils d'analyse de sentiment en attribuant le poids le plus élevé à l'algorithme le plus cohérent.

Notre algorithme apprend les pondérations des règles en fonction de leurs degrés de cohérence et d'incohérence en s'appuyant sur le mécanisme d'explication d'inférence logique.

- **Un algorithm d'instanciation efficace**

  Nous avons proposé un nouvel algorithme d'instanciation de règles, inspiré du travail de [4], qui permet de réduire le nombre d'instanciation de règles. La nouveauté de cet algorithme est de faire l'instanciation en saturant la base de connaissances, ce qui évite un grand nombre d'instanciations non nécessaires.

Le travail de cette thèse a donné lieux au publications scientifiques suivantes:

1. Wissam Maamar kouadri, Mourad Ouziri, Salima Benbenrnou, Karima Echihabi, Themis Palpanas, Iheb Ben Amor Quality of Sentiment Analysis Tools: The Reasons of Inconsistency PVLDB 2020

2. Wissam Maamar kouadri,Mourad Ouziri,Salima Benbernou,Themis Palpanas, Karima Echihabi, Iheb Ben Amor Quality of Sentiment Analysis Tools: The Reasons of Inconsistency (extended abstract) 37ème Conférence sur la Gestion de Données – Principles, Technologies et Applications

3. Wissam Maamar kouadri,Salima Benbernou, Mourad Ouziri, Themis Palpanas, Iheb Ben Amor SentiQ: A Probabilistic Logic Approach to Enhance Sentiment Analysis Tool Quality Wisdom@KDD 2020

4. Wissam Maamar kouadri, Salima Benbernou, Mourad Ouziri, Themis Palpanas, Iheb Ben Amor SentiQ: Une approche logique-probabiliste pour une qualité d'extraction du sentiment (EGC 2021)

Les travaux suivants sont sous relecture:

1. SA-Q: A System for Sentiment Analysis Quality.

2. WSSA: Weakly Supervised Semantic-Based Sentiment Analysis

Dans la suite, nous allons présenter dans le chapitre 1 une introduction pour définir le contexte et la motivation de cette thèse.

Dans le chapitre 2, nous présentons l'état de l'art et les définitions relatives aux problèmes de la qualité de données dans l'analyse de sentiment. Nous présentons une relecture pour les outils d'analyse de sentiment. Puis, nous analysons les travaux qui traitent les problèmes d'incohérence dans les outils d'analyse de sentiment ainsi que les frameworks de résolution d'incohérence dans la littérature. Enfin, nous présentons les préliminaires et définitions sur les frameworks d'apprentissage relationnel statistique.

Dans le chapitre 3, nous présentons une étude empirique profonde sur la qualité des outils d'analyse de sentiment, définissons les raisons d'incohérences, et proposons des recommandations pour sélectionner les outils d'analyse de sentiment étant donné un scénario.

Dans le chapitre 4, nous étudions l'effet de résoudre les incohérences sur la précision des outils d'analyse de sentiment. Nous présentons les détails de SAQ, une approche qui permet de résoudre les deux types d'incohérences. Puis, nous présentons les expérimentations que nous avons menées pour évaluer l'effet de résoudre les incohérences sur les performances des outils.

Dans le chapitre 5, nous présentons les détails de WSSA , notre approche faiblement supervisée pour l'analyse de sentiment, nous présentons la motivation de WSSA, ses différents modules et son évaluation expérimentale.

Finalement, dans le chapitre 6, nous présentons la conclusion,les perspectives et les directions de recherche ouverte par cette thèse.

Part I

Introduction

Chapter 1

Introduction

With the growing popularity of social media, the internet is replete with sentiment-rich data like reviews, comments, and ratings. A sentiment analysis tool automates the process of extracting sentiments from a massive volume of data by identifying an opinion and deriving its polarity, i.e., Positive, Negative, or Neutral.

In the last decade, the topic of sentiment analysis has flourished in the research community [5]–[14], and has attracted the attention of the data management community, which has studied problems related to polarity aggregation, sentiment correlation, fact-checking, and others [14]–[26].

Therefore, organizations are showing great interest in adopting sentiment analysis tools to exploit this data for decision making. For instance, it is used in politics for predicting election results, in marketing to measure user's satisfaction [19], in crowdsourcing to measure workers' relevance [27], and in the health care domain [28].

Nevertheless, sentiment analysis is still a challenging task due to the complexity and variety of natural language, through which the same idea can be expressed and interpreted using different words. Let us illustrate this issue by considering two texts (documents) expressed differently, but having the same meaning: *(1) China urges the US to stop its unjustifiable crackdown on Huawei*; and (2) *China slams United States over unreasonable crackdown on Huawei.* We notice that although the two documents are structured differently, they are, in fact, semantically equivalent paraphrases because they convey the same meaning.

The sentiment analysis research community has adopted the consensus that semantically equivalent texts should have the same sentiment polarity[3], [29]–[33]. For example, [3] proposed an approach that learns the polarity of affective events in a narrative text based on weakly-supervised labels, where the semantically equivalent pairs (event/effect) got the same polarity, and opposite pairs (event /effect) got opposite polarities. Authors in [7] have extended their dataset with opinion paraphrases by labeling the generated paraphrases with polarity labels of the original text.

However, recent works show that sentiment analysis systems assign different po-

larity labels to semantically equivalent documents, i.e., those systems are predicting different outputs for semantically equivalent inputs [1], [34]–[39]. Such documents are called *adversarial examples*. Moreover, we notice that different sentiment analysis tools attribute different polarities to the same document. The following example illustrates this problem.

1.1 Motivating Example

Our research work is motivated by a real observation from social media. We present here a data sample collected from Twitter about Trump's restrictions on Chinese technology.

Let $D = \{d_1, \ldots, d_8\}$ be a sample set of documents/tweets such that:

- d_1: Chinese technological investment is the next target in Trump's crackdown.
- d_2: Chinese technological investment in the US is the next target in Trump's crackdown.
- d_3: China urges end to United States crackdown on Huawei.
- d_4: China slams United States over unreasonable crackdown on Huawei.
- d_5: China urges the US to stop its unjustifiable crackdown on Huawei.
- d_6: Trump softens stance on China technology crackdown.
- d_7: Trump drops new restrictions on China investment.
- d_8: Donald Trump softens tone on Chinese investments.

Note that D is constructed with subsets of semantically equivalent documents. For instance, d_1 and d_2 are semantically equivalent as they both express the idea that the US is restricting Chinese technological investments. We denote this set by A_1 and we write: $A_1 = \{d_1, d_2\}$ and $A_2 = \{d_3, d_4, d_5\}$, which express that the Chinese government demands the US to stop the crackdown on Huawei, and $A_3 = \{d_6, \ldots, d_8\}$ which convey the idea that Trump reduces restrictions on Chinese investments.

A_i are partitions of D, that is: $D = \bigcup_{i=1}^{n} A_i$ and $\bigcap_{i=1}^{n} A_i = \emptyset$. We analyze D using four sentiment analysis tools: The Stanford Sentiment Treebank [40], Senticnet5 [29], Sentiwordnet [41] and Vader [8]. We associate to those sentiment analysis tools the following polarity functions respectively: P_{rec_nn}, $P_{senticnet}$, $P_{sentiwordnet}$, P_{vader}, and P_h is associated to human annotations as a ground truth. Table 1.1 summarizes the results of the analysis.

Note that different tools attribute different sentiment labels for the same document (for e.g., only P_{rec_nn} attributes the correct label to d_3 in A_2) and the same tool can

A_i	Id	P_{rec_nn}	$P_{senticnet}$	$P_{sentiwordnet}$	P_{vader}	P_h
A_1	d_1	Neutral	Negative	Negative	Neutral	Negative
	d_2	Negative	Negative	Negative	Neutral	Negative
A_2	d_3	Negative	Positive	Positive	Neutral	Negative
	d_4	Negative	Positive	Negative	Neutral	Negative
	d_5	Negative	Positive	Negative	Neutral	Negative
A_3	d_6	Neutral	Positive	Positive	Neutral	Positive
	d_7	Negative	Negative	Positive	Neutral	Positive
	d_8	Neutral	Positive	Negative	Neutral	Positive

Table 1.1: Predicted polarity on dataset D by different tools

attribute different labels for semantically equivalent documents (e.g., $P_{senticnet}$ considers d_6 as Positive and d_7 as Negative).

These inconsistencies put the quality of sentiment analysis tools in a question mark as their presence means that at least one tool has extracted an erroneous polarity. Consequently, returning an incorrect polarity in the query can be misleading and breeds a poor business decision.

To address the problem of sentiment analysis tools quality, we started by investigating the presence of inconsistencies on sentiment analysis tools, questioning whether inconsistencies are frequent, and searching for their principal causes. After that, we asked ourselves whether resolving those inconsistencies would affect tools' performances. The following section presents the main contributions we made to answer these interrogations.

1.2 Contributions

Motivated by the previous example, we study the behavior of sentiment analysis tools toward inconsistencies and reveal their causes. Further, we study the effect of resolving the two types of inconsistency on tools performances. Then based on our findings and inspired by [2], [3], we propose a new method for sentiment analysis that uses the paradigm of weak supervision and resolves both types of inconsistencies.

In summary, we made the following contributions:

1. **Algorithmic templates**

   We review the most used methods for sentiment analysis and provide an in-depth analysis of tools. Then, we define algorithmic templates for each method's category and pick up six representative tools to evaluate.

2. **Empirical evaluation**

   We conduct an empirical study that quantifies and explains the non-robustness of sentiment analysis tools in the presence of adversarial examples.

Our evaluation uncovers two different anomalies that in sentiment analysis tools: *intra-tool inconsistency*, which is the prediction of different polarities for semantically equivalent documents by the same tool, and *inter-tool inconsistency*, which is the prediction of different polarities for semantically equivalent documents across different tools. In our in-depth analysis, we offer a detailed explanation of such inconsistencies.

Existing works on [1], [34], [38], [39], [42] have typically focused on generating data for adversarial training and for debugging models [34]. Instead, our work focuses on studying the problem from a data quality point of view, and understanding the reasons behind the aforementioned inconsistencies.

To the best of our knowledge, we are the first to study the consistency of sentiment analysis tools on the sentence level. The only work studying polarity inconsistency [43] does this at the word level by checking the polarity consistency for sentiment words inside and across dictionaries.

Further, we evaluate six state-of-the-art sentiment analysis tools (implementing two tools and get the codes of other tools) and define two new metrics to evaluate the inconsistency. Our evaluation focuses on four axes:

• Statistical: The goal of the experiments is to evaluate whether the inconsistencies are rare anomalies or not, find out the types of anomalies, and finally, determine the tools and datasets that have the most anomalies. The findings of this experiment reveal that the inconsistencies are frequent events and occur on different tools and datasets.

• Structural: The purpose of the experiments is to verify the impact of analogical set structures on intra-tool inconsistency by checking if inconsistencies are due to the semantic or syntactic distance between documents. In other words, we verify whether inconsistencies depend more on syntax difference between sentences or on the semantic difference between them. The findings show that tools are very sensitive to the syntax of the text.

• Semantic: The purpose of the experiments is to verify whether the inconsistencies are frequent between polar facts or opinionated documents and if minor inconsistency implies a higher accuracy. In other words, we evaluate the efficiency of tools in terms of accuracy and inconsistency. The findings have shown that we have more inconsistencies in polar facts and an inverse correlation between inter-tool inconsistency and accuracy.

• Machine learning hyperparameters, accuracy, and inconsistency: In this experiment, we focus on studying the impact of the learning hyper-parameters on intra-tool inconsistency and the accuracy of the CNN [44]. We find that inconsistencies are present in different learning configurations. Moreover, we pinpoint configurations that reduce inconsistency and guarantee accuracy.

3. **Bench-marking**

To evaluate the inconsistencies, we built a benchmark containing paraphrases by extending five publicly available datasets using the syntactically controlled

paraphrase networks (SCPNs) [1]. To the best of our knowledge, no such dataset exists publicly.

We have chosen five datasets to build the benchmark based on their popularity [10], [44]–[50] and size. Then, we augmented them with paraphrases using the syntactically controlled paraphrase networks (SCPNs) tool.

To improve the quality of the benchmark by reducing the produced error rate, we propose a protocol that minimizes the human effort and the cost of data quality verification.

Tips and recommendations

Based on our evaluation findings, we suggest a set of tips and recommendations to select the appropriate tools for a given scenario.

4. **Framework for intra-tool and inter-tool inconsistency resolution**

   Many works have resolved inconsistencies to improve systems' accuracy. For instance, the work in [2] has considered various labeling functions and minimized the inter-tool inconsistency between them based on different factors. In addition, the work [3] has proposed to create a corpus of (event/effect) pairs for sentiment analysis by minimizing the sentiment distance between semantically equivalent (event/effect) pairs. However, those works studied the two types of inconsistency independently, and to the best of our knowledge, no previous work has studied the effect of solving the two types of inconsistency (intra-tool and inter-tool) on accuracy.

   Hence, we propose SAQ (Sentiment Analysis Quality), a framework for inconsistency resolution in sentiment analysis that resolves intra-tool and inter-tool inconsistencies.

   SAQ uses the Markov logic network to model the documents, tools, and relations between them. The intuition of SAQ is to seek convergence to the golden truth by resolving intra-tool and inter-tool inconsistencies based on the two consensuses: a document has a unique polarity, and the semantically equivalent documents have the same polarities.

5. **Study on the impact inconsistency resolution**

   We study the impact of resolving the two types of inconsistency (intra-tool and inter-tool), focusing on the improvement we can obtain.

   We analyze the obtained improvement based on the number of tools and their primary accuracy. We compare the accuracy improvement obtained by SAQ to three conventional state-of-the-art methods for resolving inconsistency using the benchmark we created and discuss the results to define the necessary conditions to obtain improvement.

   The evaluation results of SAQ are promising and prove that we obtain better improvement when resolving the two types of inconsistency (an improvement of 20% accuracy).

6. **WSSA: Weakly Supervised Semantic-Based Sentiment Analysis**

   We propose WSSA, the first weakly supervised sentiment analysis method, which considers the two types of inconsistency with domain knowledge to perform weak supervision.

   Our approach considers a set of sentiment analysis tools that provide noisy polarity labels (weak labels), then aggregates them by resolves the two types of inconsistency. The aggregation process in WSSA is guided by domain expert knowledge and preference order set between documents based on their subjectivity.

   To ensure the scalability of WSSA, we used the Probabilistic Soft Logic (PSL) [51]. This scalable statistical relational learning framework allows the modeling of tools, documents, and relations between them. Moreover, it allows integrating the domain expert knowledge and representing the inconsistency and the uncertainty.

7. **Unsupervised Approach for Learning Weight**

   Since learning in PSL is supervised and needs to have a ground truth of polarities, we propose a novel unsupervised logic-based approach to learn tools' weights through logical entailment explanation.

   Our learning algorithm ranks tools based on their consistency, which attributes higher weight to the most consistent tools.

8. **Efficient grounding algorithms**

   We propose a new grounding algorithm inspired by the work [4] that reduces the number of grounded rules. The novelty of our grounding algorithm is to fully entail the implicit knowledge then grounds rules based on the fully entailed knowledge Base, which avoids unnecessary grounding.

1.3 Publications

Published

- Wissam Maamar kouadri, Mourad Ouziri, Salima Benbenrnou, Karima Echihabi, Themis Palpanas, Iheb Ben Amor Quality of Sentiment Analysis Tools: The Reasons of Inconsistency PVLDB 2020
- Wissam Maamar kouadri,Mourad Ouziri,Salima Benbernou,Themis Palpanas, Karima Echihabi, Iheb Ben Amor Quality of Sentiment Analysis Tools: The Reasons of Inconsistency (extended abstract) 37ème Conférence sur la Gestion de Données – Principles, Technologies et Applications
- Wissam Maamar kouadri,Salima Benbernou, Mourad Ouziri, Themis Palpanas, Iheb Ben Amor SentiQ: A Probabilistic Logic Approach to Enhance Sentiment Analysis Tool Quality Wisdom@KDD 2020

- Wissam Maamar kouadri, Salima Benbernou, Mourad Ouziri, Themis Palpanas, Iheb Ben Amor SentiQ: Une approche logique-probabiliste pour une qualité d'extraction du sentiment (EGC 2021)

Under submissions

- SA-Q: A System for Sentiment Analysis Quality.
- WSSA: Weakly Supervised Semantic-Based Sentiment Analysis

1.4 Thesis Outline

In Chapter 2, we present state-of-the-art and preliminaries related to the problem of sentiment analysis quality.

We review sentiment analysis tools, analyze the studies that handle the problem of polarity inconsistency in the literature, and review the frameworks for inconsistency resolution methods.

Finally, we present the preliminaries represented by the logical and statistical relational learning backgrounds.

In Chapter 3, we present an in-depth empirical study on sentiment analysis tools inconsistency, define the reasons behind the inconsistency, and propose tips and recommendations to choose a tool under a given scenario based on the analysis.

In Chapter 4, we study the effect of resolving the two types of inconsistency on sentiment analysis tools. First, we present the details of SAQ, a framework that we propose for resolving inconsistencies in sentiment analysis tools. Then we present the experiments we lead to evaluate the effect of inconsistency resolution on tools performances.

In Chapter 5, we present WSSA, our weakly-supervised approach for sentiment analysis. We present the motivation behind WSSA, its different modules, and the experimental evaluation.

Finally, we conclude in Chapter 6 and discuss open research directions.

Part II

State-of-The-Art

Chapter 2

State-of-The-Art and Preliminaries

The quality of sentiment analysis tools is a problem involving different research axes such as sentiment analysis and inconsistency resolution methods.

Before presenting the main contributions of the thesis, we present in this chapter an overview of sentiment analysis (Section 1), followed by a review of inconsistency resolution methods in the literature (Section 2). Then we present works that have used inconsistency resolution to build powerful frameworks for classification (Section 3).

Finally, we present the preliminaries and the definitions about SRL (statistic relational learning) and description logic that we used in our work (Section 4).

2.1 Sentiment Analysis Tools

Sentiment analysis is the process of automatically extracting the polarity from a document d that could be a review, a comment, or a tweet.

Earlier work on sentiment analysis has focused on creating word lexicons of polar words [8], [52], [53] or using the traditional statistical methods with features engineering (SVM, naive Bayes, KNN) similar to the methods in [54]–[57]. These methods learn the function F that extract the polarity Y from the features vector X extracted from the document d .i.e: $Y = F(X)$

However, those methods have low accuracy, and they are inefficient with polar facts [58]. Recent works are using more sophisticated algorithms (such as deep learning [36], [47], [59]–[61]) to enhance the accuracy and overcome the limits of the previous methods.

In this section, we define sentiment analysis formally, present the details of the most used state-of-the-art methods and define the algorithmic templates of those methods.

2.1.1 Definitions

Definition 1. *Sentiment Analysis*

It is the process of extracting the quintuplet $< E_i, F_{ij}, H_z, O_{ijz}, t >$ from a document d, such that: E_i is an entity i mentioned in document d, F_{ij} is the j^{th} feature of the entity i, H_z is the holder or the person who gives the opinion, Ω_{ijz} is the polarity of the feature ij at the time t [62]. Notice that the same document can contain several entities and that each entity can have a several features.

Definition 2. *Polarity*

We call a polarity Ω the semantic orientation of the sentiment in the text. The polarity can be Negative, Neutral or Positive and is denoted by $\Omega \in \Pi$ and $\Pi = \{Negative, Neutral, Positive\}$.

Many researches model the polarity as a scalar in a continues interval divided into three sub-intervals where each one represents a polarity orientation. formally:

$$\Omega \in [l_{neg}, l_{pos}]\, s.t\, [l_{neg}, l_{pos}] = [l_{neg}, l_{neut-}] \cup [l_{neut-}, l_{neut+}] \cup [l_{neut+}, l_{pos}]$$

In some studies this interval is divided to more fragments like in [40]. In this case, polarity values express both the strength of the polarity and its semantic orientation. In other works, such as in [3], they represent the polarity by a vector V of three components s.t: $V = [w_{neg}, w_{neut}, w_{pos}]$, where each component w_i represents the semantic orientation and the strength of the polarity.

Definition 3. *Polarity function*

For each sentiment analysis tool t_k, we associate a polarity function P_{t_k} defined as: $\forall P_{t_k} \in \Gamma,\ P_{t_k} : D \rightarrow \Pi$, with Γ the set of all polarity functions and D a set of documents.

Definition 4. *Ground truth*

We denote by P_h the polarity given by human annotation that we consider the ground truth.

Definition 5. *Polar Word*

We define a polar word as a word that has a polarity different from Neutral ($P_h(w_j) \neq$ *"Neutral"*).

Definition 6. *Opinionated Document*

Let's consider a document d_i as a set of words such that $d_i = \{w_1, \ldots, w_m\}$. An opinionated document is a document that contains a polar word, formally: $\exists\ 1 \leq j \leq m$ *and* $w_j \in d_i$, *such that* $P_h(w_j) \neq$ *"Neutral"*.

Definition 7. *Polar Fact/Objective Document*

A polar fact, a.k.a an objective document, is a document that does not contain a polar word, i.e., d_i is a polar fact iff: $\forall w_j \in d_i, such\ that\ 1 \leq j \leq m,\ P_h(w_j) = "Neutral"$ and $P_h(d_i) \neq "Neutral"$. Notice that the polarity in polar facts could depend to the context of the sentence. In our work, we focus on the common-sense context.

2.1.2 Lexicon-Based Approaches

This family of approaches is considered a trivial solution to sentiment analysis. It encompasses unsupervised methods that use a word matching scheme to find the sentiment of a document based on a predefined dictionary of polar words [52] [8], [53], concepts [29], [63], [64], or events/ effects [3], [65], [66]. We propose a description of the template followed by most lexicon-based methods in Algorithm 1.

In this category of methods, we represent the document as a bag of words: splitting it into words, stemming them, and then suppressing all punctuation and stop words. After that, we search each word's polarity score in the dictionary and consider the sentence's polarity as the aggregation of different words' polarities (mean in this case).

We run Algorithm 1 by using Sentiwordnet [41] as a lexicon on document d_8 from Example 1.1. First, we represent the document in a bag-of-words model to tokenize it, **(step 1)**: Donald:1, Trump:1, soften:1, tone:1, on:1, Chinese:1, investments:1. where the numbers represent the frequency of tokens in the sentence. Then we search the sentiment score of each token in the dictionary, **(step 2)** we find: $list = \{0, 0, 0, -0.125, 0, 0, 0\}$ (we multiply the polarity by the effective of the token). We aggregate tokens' polarities, by calculating the mean of the polarity scores **(step 3)**, and get the polarity label associated with these scores **(step 4)**, we find that: $P_{sentiwordnet}(d_8) = Negative$.

2.1.3 Rule-Based Approaches

Rule-based methods [8], [29] are similar to lexicon-based methods with an additional inference layer that allows deducing the correct polarity of documents based on linguistic rules.

First, we create a lexicon of words, concepts, or events/effects. Then, a layer of syntactic and part-of-speech rules is added to allow the inference of the correct polarity label. For example, the Vader tool [8] uses the following syntactic rules:

- If all characters of a polar word are in upper case (ex: GOOD), the polarity intensity increases by 0.733.
- The polarity intensity decreases in text preceding the conjunction "but" by 50% and increases in the text that follows by 50%.

Algorithm 1 Lexicon-Based Sentiment Analysis

Input : d_i : *Document*
Output : Polarity label $l_i \in \Pi$

procedure LexiconBased
  //**step1:***Tokenize the document*
    $W = tokenize(d_i)$
  //**step2:** *Calculate the polarity of each token in* W
    for each $w_i \in W$:
      $list.add(P_*(w_i))$
  //**step3:***Aggregate words polarities (using mean*
  // in most cases)
    $score = mean(list)$
  //**step 4:***Get the polarity label based on*
  //the polarity score
    $l = getLabel(score)$
  return l

Senticnet [29] is a semantic network of concepts (lexicon with relations between concepts) that has a layer for polarity inference. It uses linguistic rules named "sentic patterns" to infer the correct polarity of the sentence based on its structure. As an example: if the concept is in a polarity switcher's scope, such as negation, its polarity will be reversed from Positive to Negative. We propose the Algorithm 2 that represents the general template adopted by most rule-based methods.

Let us consider the document "China does not want to supplant the US, but it will keep growing" that we analyze using Senticnet:**(step 1)**, Senticnet divides this sentence into concepts using the *tokenize* function, which produces the list W of tokens (a.k.a. concepts):

$$W = \{china, supplant\_US, keep\_grow\}$$

(step 2) Senticnet determines the polarity of each concept as a list of couples $< Concept : polarity >$:

$$list = \{china : Neutral\,, supplant\_US : Positive\,, keep\_grow : Positive\}$$

(step 3) The function $infer\_polarity$ activates Sentic patterns to infer each concept's polarity. The concept $< Supplant\_\ US >$ is in the scope of a polarity switching operator, which inverts the polarity of this concept to "Negative".

(step 4) When aggregating the concepts polarities, Senticnet encounters the conjunction $< but >$; therefore, it infers the polarity as being Positive because it applies the following adversary rule defined in Sentic patterns: the polarity of the document is equivalent to the polarity of the text following the conjunction $< but >$.

Algorithm 2 Rule-Based Sentiment Analysis

Input: d_i: Document
Output: Polarity label $l_i \in \Pi$

```
procedure RuleBased
    // Step1: Split the sentence into units(words,
    //concepts, event/effect )
        W = tokenize(d_i)
    // Step2: Get the polarity of each token in W from
    //the semantic network/lexicon
        for each w_i ∈ W :
            list.add(P_*(w_i))
    // Step3: Infer the final polarity of tokens using
    //the set of defined syntactic rules s
        score = infer_polarity(list, s)
    // Step5 : Get the polarity label based on the
    //polarity score
        l = getLabel(score)
    return l
```

2.1.4 Learning-Based Approaches

Researchers have recently explored deep learning models [40], [47], [60], [67]–[69] in sentiment analysis to exploit its ability to extract the relevant features for sentiment analysis from the text (instead of the feature engineering). For instance, [40] has proposed a Recursive Neural Network model (RecNN) for sentiment classification. These methods learn a vector representation of sentences (sentence embedding) through a recursive compositional computation, then feed it to a Softmax sentiment classifier.

On the embedding level, a standard Convolutional Neural Network (Text_CNN) was proposed in [44] for text classification tasks, including sentiment analysis. It uses a word embedding model pre-trained on the Google News corpus [70], which guarantees a good coverage of the language space and catches the semantics of the sentence.

Authors in [60] have refined the sentence embedding level and proposed a character to sentence CNN for short text classification (Char_CNN) with two convolutional layers that handle sentiment features at character and sentence levels (word order, character characteristics).

Authors in[71] have proposed a convolutional recurrent neural network for text classification, including sentiment analysis. They have used the bi-recurrent neural network to learn the word representation; then, for each word, they have considered its context to capture the contextual information. Moreover, the model can reserve a more extensive range of word order when learning representations of texts.

We propose the descriptions of the methods that use CNNs and follow the general

Algorithm 3 (CNN for Sentiment Analysis (training)

Input :- training dataset $D = \{d_1, \ldots, d_n\}$
-L: Hyper parameters list//batch size,
window size,number of filters ...
Output: Trained model

1: **procedure** Training
2: //**Step1**: initialize weights
3: //**Step2**: Document representation
4: //(word embedding)
5: **Foreach** $d_i \in D$:
6: $U_i = \{u_{ij} | \forall w_{ij} \in d_i, u_{ij} = g(w_i)\}$
7: $list.add(U_i)$
8: // **Step4**: Training network
9: **For** nb_epoch **do** :
10: Convolution: Equation 2.1
11: Pooling
12: Classifier
13: Optimization (backpropagation) Equation 2.2

template described in Algorithms 3 for training and 4 for classification. CNNs perform document compositionality and transform the matrix document into a vector of features used by a classifier. Thus, in this case, we first split the document d_i into a sequence of n words such that

$$d_i = \{w_{i1}, \ldots, w_{in}\}$$

Then, we represent each word as a vector to obtain a matrix representation of the document. Many research works [44], [60], [67], [71] have used a word embedding model for this task. Other methods [70], [72] use a pre-trained model to initialize the word embedding vectors while others initialize the vectors randomly. In this case, our document d_i will be represented as:

$$d_i = \{U_{i1}, \ldots, U_{in}\}$$

Where $U_{ij} \in \mathbb{R}^k$, k is the size of the embedding vector , $U_{ij} = g(w_{ij})$, and g is a word embedding function. We concatenate vectors to obtain the matrix representation of the document as follows:

$$M_i = U_{i1} \oplus U_{i2} \oplus \cdots \oplus U_{in} M_i \in (\mathbb{R})^{\mathrm{k} \times |\mathrm{d_i}|}$$

This matrix is the input of the convolution layer, which extracts features from the matrix M_i by applying convolutions between filters W and the input matrix on a

Algorithm 4 CNN for Sentiment Analysis (Classification)

Input : d_i : Document, $model$
Output : Polarity label $l \in \Pi$

1: **procedure** LearningBasedPrediction
2: $load(model)$
3: Convolution: Equation 2.1
4: Pooling
5: $v = Classifier//$ returns a value v
6: **return** get_polarity(v)

window of size l. The convolution operation is defined by:

$$z_j = h(W.M_i[j, j + l - 1] + b) \tag{2.1}$$

where h is the activation function that could be $tanh$, $ReLU$,or $Softmax$, z_j is the features map (resulted features), $W \in \mathbb{R}^{l \times o}$ where l and o are hyper-parameters. Typically, $o = |d_i|$. W is the filter to learn and b is the bias. Pooling is applied to extract only the significant features. After that, the extracted vector of features is fed into a normal classifier (Neural Network, SVM, or Softmax). The filters are the parameters to learn. The training process consists of using forward-propagation then updating weights using stochastic back-propagation with the objective of maximizing the log likelihood[1].

$$max \sum_{i=1}^{n} log\ pr(P_*(d_i)|d_i, \theta) \tag{2.2}$$

Despite the efficiency of deep learning methods for sentiment analysis, recent works show that those tools assign different polarity labels to equivalent documents, i.e., those systems are predicting different outputs for equivalent inputs that should have the same polarity [1], [34]–[39]. Such documents are called *adversarial examples*.

Moreover, sentiment analysis tools depend strongly on the training dataset, leading to different outputs for the same input when changing the training set, making the quality of sentiment analysis a question mark.

The following section presents an overview of the work that studies sentiment analysis quality in the literature.

2.2 Sentiment Analysis Quality in Literature

The problem of inconsistency in sentiment analysis has attracted the interest of researchers in the data management community [15], [21], [24]. Studies performed on

[1]Equivalently, some methods [60] minimize the Negative likelihood.

this problem can be classified into two categories: time-level inconsistency (or sentiment diversity), and polarity inconsistency represented by both intra-tool and inter-tool inconsistencies.

For instance, authors in [24] have studied the inconsistency of opinion polarities (on a specific subject) in a given time window, as well as its evolution over time, and propose a tree-based method that aggregates opinion polarities by considering their disagreements at large scale. Authors in [15] proposed a set of hybrid machine learning methods that rank tweet events based on their controversial score (inconsistency score) in a time window.

As an example of polarity inconsistency work, authors in [21], [43] have studied the polarity consistency on sentiment analysis dictionaries on two levels (intra- and inter-dictionaries). Hence, their study has focused only on one type of sentiment analysis tools (lexicon-based methods with word dictionary), while our study is more thorough and includes a wider range of algorithms (rule-based, machine learning-based and concept lexicon methods) and data.

Other studies about intra-tool inconsistency have been limited on earlier research that studied sentiment preservation in translated text using lexicon-based methods [73], [74]. The findings of [73] have shown the consistency of their lexicon based method predictions regarding different languages (or different writing of the document).

Inconsistencies in the sentiment analysis tools are costly and need to be resolved for tools quality improvement. The following section presents a detailed overview of inconsistency resolution methods from the literature that improves the quality of the frameworks.

2.3 Inconsistency Resolution Methods and Limits

Several research fields have resolved the inconsistency to enhance the frameworks' performances, such as ensemble methods, weak supervision, and truth inference in crowdsourcing.

2.3.1 Ensemble Methods

Ensemble methods [75]–[77] represent the earlier efforts to improve classification performances by resolving inconsistencies between weak classifiers. They are a category of learning algorithms that build a set of classifiers and then use their predictions' weighted vote to classify new data points.

Bayesian averaging [76] is the original ensemble method. It makes predictions using an average over several models weighted by their posterior probabilities.

Bagging [77] considers a set of classifiers trained with data sampled from the training set (sampling by replacement). After that, it aggregates the inconsistent predictions

given by models by using majority voting over the classification results.

Boosting [78] algorithms consider a set of classifiers weighted based on their prediction errors on the training step where high weight is assigned to the classifier that makes fewer errors. It aggregates the inconsistent (different) predictions given by the weak classifiers using a weighted majority voting in the prediction step.

2.3.2 Truth Inference Methods

The wide use of deep learning methods and their skills in different classification tasks has augmented the need of having annotated data. However, since expert data is costly, crowdsourcing is a cheaper and faster solution to annotate data and address complex problems for the machine [79]–[81]. In crowdsourcing, we give a task to human workers to decide the task's answer. For instance, we give a sentence to a worker, and we ask him to attribute a polarity positive, negative or neutral for the sentence.

Due to the expense of crowdsourcing, workers may be of low quality and cannot be wholly trusted to provide reliable labels. Thus it is crucial to control the quality of crowdsourcing. To deal with this problem, most approaches assign the same task to several workers then aggregate the answers to infer the correct answer (Truth). Truth inference in crowdsourcing[82]–[87] is a widely studied problem in the literature that infers the correct answer for a task based on answers' redundancy. A typical truth inference method learns workers' quality and infers the truth based on the learned quality.

Following the study in [88], truth inference methods could be classified as optimization [89]–[91], probabilistic graphical model [92]–[94], and direct computation methods.

The direct computation methods are represented by majority voting, representing the straightforward method for truth inference and inconsistency resolution by considering the true answer as the most repeated answer by workers. However, this method considers all workers with the same quality, which is not always the case and may lead to errors in truth inference when assigning the same weight to workers with different qualities. Therefore, the optimization and the probabilistic methods that learn workers' quality.

The intuition of the optimization-based frameworks is to define an optimization function that models the relation between workers' quality and the truth. Then, iteratively derives the quality of workers and the truth (the correct answer). It minimizes the sum distance between workers' answers and the inferred truth, i.e., minimizing the inconsistency between workers. Formally:

$$\min_{q^w, v_i^*} \sum_{w \in \omega} q^w \sum_{t_i \in \mathcal{T}}^{w} d(v_i^w, v_i^*)$$

With q^w the worker's w quality, v_i^* the inferred truth for the i^{th} task, ω the set of

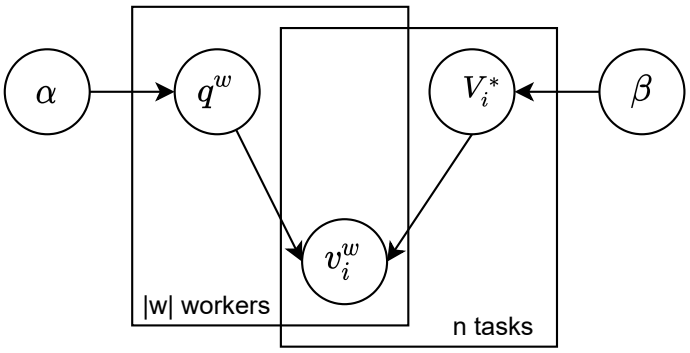


Figure 2.1: Probabilistic Graphical Model [88] with q^w workers quality, v_i^w workers answers , and v_i^* the truth.

workers, and v_i^w the workers w answer for the task t_i.

Most methods inferred in step j the worker's quality based on the truth inferred in step j-1, then they repeat the process until convergence.

Many research works [92]–[94] have used probabilistic graphical models to represent the conditional dependencies between worker's quality, worker's answer, and the truth. Figure 2.1 shows an example of the graphical model used by most methods. Each node represents a random variable: q^w, V_i*, are latent variables that represent worker's quality and the truth respectively, α and β are priors (observed values) for the worker quality and the truth respectively, and v_i^w is a random variable that models worker's answer. The edges in the graph represents the conditional relations between the random variables.

2.3.3 Weak Supervision

Weak supervision is a new learning trend that train classifiers using a low-quality data (with uncertain labels) then aggregates the different outputs to create a stronger classifier.

Recently, the data management and data mining communities have been interested in weakly supervised approaches [2], [3], [95]–[99] and use them for different tasks.

For instance, the work [2] has used weak supervision to generate training data. First, the labels are integrated from various labeling functions; then, the generated data is used to train a discriminative model. The recent work [95] proposes a novel weak supervised approach that can be used on different tasks such as the bug detection task. This method uses a combined set of noisy labeled and unlabeled data then evaluates the noisy data's risk on the model.

Furthermore, the work [96] has proposed a weakly supervised method that uses the redundancy and the noise in data for relational entity ranking. The work in [100] has proposed a weakly supervised method for fake news detection. The weak labels come

from a weak fake news classifier then passed to a reinforcement learning selector that selects samples for training that maximize the accuracy on a small labeled dataset.

Several research works [3], [97]–[99] have applied weak supervision to sentiment analysis, especially with the massive existing noisy data and the difficulty of obtaining high-quality labeled data.

For instance, the work [97] proposes a novel deep learning framework for sentiment analysis on the reviews by considering noisy labels from ratings. In this method, the noise is leveraged through a new embedding approach that supposes the distance between the same label sentences is lower than between sentences with different labels.

Further, the work [99] proposes a method for sentence-level sentiment analysis by considering as weak labels several polarity sources such as the document containing the sentence, the polarity aggregation of sentence's words, and the relation between sentences; then learning the closest polarity of these sources. Finally, the work [3] proposes a weakly supervised method for effective event classification by reducing the inconsistency between the semantically equivalent events.

As we can observe, the principle intuition in all methods is to use redundancy to reduce the noise, which creates powerful tools. However, to the best of our knowledge, no previous work has considered redundancies from documents and tools by analyzing the semantic equivalence between documents with weak labels from different sources.

Hence in the following chapters, we study the usefulness of using both the redundancy from tools by considering several weak sources and the redundancy from the side of the documents by using the semantic equivalence.

Before presenting the main contributions of the thesis, the following section introduces the fundamentals and recurring concepts that will later appear in the following chapters.

2.3.4 Logical Background

Definition 8. *Knowledge base*

In our setting, a knowledge base is defined as $\mathcal{KB} =< \mathcal{R}, \mathcal{F} >$, where $\mathcal{R}$ is a set of rules (FOL formulas) defining the vocabulary of our application (concepts and relations between them) and $\mathcal{F}$ is a set of facts representing the instances of the concepts or individuals defined in $\mathcal{R}$.

Definition 9. *Rule*

An inference rule $R \in \mathcal{R}$ of a knowledge base $\mathcal{KB}$ has the form $R : H \leftarrow B$, where H is called the head or conclusion of the rule. The head is a disjunction of literals such that $H = H_1 \vee H_2 \vee .. \vee H_n$.

B is called the body and is a conjunction of literals such that $B = B_1 \wedge B_2 \wedge .. \wedge B_m$,

where H_i B_j are literals expressed with predicates applied to terms (variables, constants and functions). The rule also have an equivalent notation $\bigvee_{i<n} H_i \bigvee_{j<m} B_j$.

Definition 10. *Interpretation*

An interpretation I is an assignment of meaning to the symbols of a formal language, such that $I =< \Delta, .^I >$, where Δ is an arbitrary non-empty set and I is a function that maps n-ary function symbols ($f^{\mathcal{I}}$) to function over Δ, n-ary Predicate $P^I \subseteq \Delta^n$ symbols to relation over Δ and individual constants $a^I \in \Delta$. In the rest of our work, we consider a fragment of the FOL that contains only unary and binary predicates, namely concepts and relations.

Definition 11. *Entailment*

The formula ϕ is logically implied by a formula ψ ($\psi \models \phi$), if all the models $\mathcal{I}$ of ψ are also models of ϕ, namely:

$$\forall \mathcal{I}, \mathcal{I} \models \psi \quad \Rightarrow \mathcal{I} \models \phi$$

More generally, a knowledge base KB logically entails formula F, $\mathcal{KB} \models \mathcal{F}$ if all the models of the $\mathcal{KB}$ (that evaluate $\mathcal{KB}$ to True) are also models for $\mathcal{F}$ (evaluate the formula $\mathcal{F}$ to True also).

Definition 12. *Grounding*

A logical rule is grounded out by performing all distinct substitutions from variables to constants such that the resulting ground atoms are in the base $\mathcal{F}$, $\forall R, \exists x, \exists C \mid subst(\{x/C\}, R)$, where *subst* is a function that performs the substitution. This procedure produces a set of grounded rules, which are rules containing only grounded atoms.

Definition 13. *Explanation*

The justification of the calculations consists of explaining the result returned by a logical inference algorithm by giving the elements of the knowledge base (rules and facts) that led to the inferred result. The objective being to constitute the minimum set of these elements. Formally speaking, the explanation of an entailment α from a knowledge base $\mathcal{KB}$ denoted $\mathcal{KB} \models \alpha$ is justified by the trace $\mathcal{KB}' \subseteq \mathcal{KB}$, if $\mathcal{KB}' \models \alpha$ that there is no other knowledge base $\mathcal{KB}''$ such as $\mathcal{KB}'' \subseteq \mathcal{KB}'$ and $\mathcal{KB}'' \models \alpha$. In our case we would like to explain requests for unsatisfiability and inconsistency with rules and facts from the knowledge base.

2.4 Statistical Relational Learning

The traditional modeling and query answering methods deal with exact data. However, real-life data in critical domains such as financial and medical fields is far from

being precise. Moreover, integrating data from different sources generate inconsistencies which bring the data uncertainty problem to the front. These factors motivate the development of a framework that allows the representation and the querying of uncertain data. Hence, research has combined probabilistic graphical models with relational representation to create statistical relational learning (SRL) discipline [101].

SRL methods [51], [102]–[104] define a probability distribution over the relational data: entities and relations. The basic idea is to map the relational representation (first-order logic, knowledge graph, relational model) into a probabilistic graphical model (Bayesian network or Markov random fields most of time). Then calculate (learn) the probabilistic distribution over the resulted model.

Depending on the used graphical model, we can classify statistical relational learning works into **models based on Bayesian networks** and **models based on Markov networks**. In our work we focus on models that use Markov networks.

A Markov Network (MN) [105] is an undirect probabilistic graphical model that defines the probability of variables values assignment based on the clique potential:

$$P(X = x) = \frac{1}{Z}\Pi_k \phi_k(x_{\{k\}})$$

Where x_k is the values assignment to the $k$the clique variables, Z is a normalization factor given by $\sum_{x \in X} \Pi_k \phi_k(x_{\{k\}})$, and ϕ_k is the potential function.

Markov logic network defines Markov networks based on the grounding of weighted FOL formulas where each grounded clause represents a node, and each clique represents a grounded formula. For instance, let us consider the two following FOL formulas:

$$F1 : IsPositive(d_j) \leftarrow IsPositive(d_i) \wedge SameAs(d_i, d_j)$$

$$F2 : IsNegative(d_j) \leftarrow IsNegative(d_i) \wedge SameAs(d_i, d_j)$$

and the following documents $d1$ and $d2$, instances for d_i and d_j. Figure 2.2 represents the MN associated with the two FOL formulas $F1$ and $F2$ grounded with $d1$ and $d2$. The cliques $\{IsPositive(d1)$, $IsPositive(d2)$, $SameAs(d1, d_2)\}$, and $\{IsNegative(d1)$, $IsNegative(d2)$, $SameAs(d1, d_2)\}$ represent, respectively, the grounded formulas $F1$ and $F2$ with the documents $d1$ and $d2$.

Each node supports two values, 0 or 1, corresponding to instances' truth values.

The instances $IsPositive(d2)$ and $IsNegative(d2)$ are contradicted, indicating that this world is impossible in a traditional knowledge base system where the constraints are hard. Hence the usefulness of MLN is to soften rules by assigning weights (importance degrees to formulas); therefore, when a constraint is violated, the world becomes less important but not impossible.

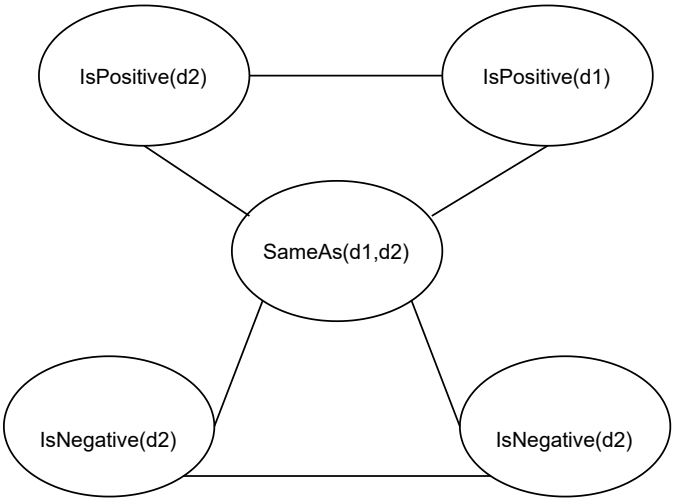


Figure 2.2: Example of generated MN from the formulas F1 and F2

2.4.1 Markov Logic Network

Definition 14. *Markov logic network (MLN)*

MLN is defined as a set of weighted first-order logic (FOL) formulas with free variables

$$L = \{(l_1, w_1), \ldots, (l_n, w_n)\}$$

with $w_i \in IR \cup \infty$ and l_i a FOL constraint. With a set of constants $C = \{c_1, \ldots, c_m\}$, it constitutes the Markov network $M_{L,C}$. The $M_{L,C}$ contains one node for each predicate's grounding whose value is 1 if the grounding is true and 0 otherwise. Hence, MLN can be considered as a template for MN.

Definition 15. *Hard and Soft Constraints*

Hard constraints are constraints with infinite weights: $w_i = \infty$. A world x that violates these constraints is impossible. **Soft constraints** are constraints with a finite weight ($w_i \in IR$) that can be violated.

Definition 16. *World's Probability*

A **world** x over a domain C is a set of possible grounding of MLN constraints over C, **world's probability** is the probability distribution of possible worlds x in $M_{L,C}$ given by

$$Pr(X = x) = \frac{1}{Z} exp(\sum_i w_i, n_i(x))$$

where $n_i(x)$ is the number of the true groundings of F_i in x and Z is a normalization factor.

The two primary tasks of MLN are learning the weights of rules or the structure and inference. In the next section, we present an overview of the learning and inference process.

2.4.1.1 Weights' learning

The learning algorithms in MLN come in two flavors: discriminative and generative weight learning. In generative weight learning, we maximize the likelihood or the pseudo-likelihood. In discriminative weight learning, we maximize the conditional likelihood or max-margin measures.

Generative weight learning involves inducing a global optimization of the world by optimizing the global probability of all variables. The weights can be learned by maximizing the likelihood of the knowledge base. We note that the full grounding of atoms is difficult; hence, the closed world assumption that considers all atoms not in the database false. Generative weight learning consists in finding the weights that optimize the following probability:

$$logP_w(X = x) = n_i - \sum_{x'} P_w(X = x')n_i(x')$$

With n_i is the number of the correct (true) grounding of the formula F. The gradient is simply the difference between the true grounding and the expectation given the current model to learn the rules' weights, we use the discriminate training described in [106].

In **Discriminative weight learning**, the victor's weights are discriminatively learned by maximizing the conditional log-likelihood (CLL) given by:

$$P(Y = y|X = x) = \frac{1}{Z_x} exp(\sum_i w_i n_i(x, y))$$

$n_i(x, y)$ is the number of correct grounding of the formula F_i in x and y.

Optimizing this probability consists in calculating the gradient descent. The derivative of this probability is:

$$\begin{aligned}\frac{\partial}{\partial w_i} - logP_w(Y = y|X = x) &= -n_i(x, y) + \sum_{y'} P_w(Y = y'|X = x)n_i(x, y') \\ &= E_{w,y}[n_i(x, y)] - n_i(x, y)\end{aligned}$$

Since computing $E_{w,y}$ is intractable, [107] has proposed to estimate the expectation using a MaxWalkSat algorithm to find the MAP state with a voting perception algorithm. This method may lead to errors since the MaxWalkSat algorithm in this case does not reach the global MAP state. To overcome this problem, [106] proposed to use

the scale conjugated gradient that uses the hasanian to choose the step size instead of line search used to speed up the gradient descent.

2.4.1.2 Inference

The inference in MLN [107] contains two steps: the grounding step, where we sample all possible worlds based on the priors and construct a large weighted SAT formula used in satisfiability calculation. Then, the search step finds the best weight assignment to this SAT formula.

We distinguish two types of inference: MPE inference which consists in finding the most probable explanation, and MAP inference that computes the arbitrary conditional probability.

MPE inference is a basic inference task that consists in finding the most probable state of a world Y given evidence X. Formally:

$$argmax_y P(y|x) = argmax_y \frac{1}{Z_x} exp(\sum_i w_i n_i(x, y))$$

Where $n(x, y)$ is the number of true groundings of the F_i in $x \cup y$. Hence, MPE inference finds the truth assignment that maximizes probability. We solve this problem using MaxWalkSat solver, a weighted variant of the walk-Sat local search satisfiability solver.

MAP inference consists in computing the probability that the formula F_i holds giving the $M_{L,C}$ by calculating

$$P(F_1|F_2, M_{L,C}) = \frac{P(F_1 \wedge F_2, M_{L,C})}{P(F_2|M_{L,C})}$$

The direct computation of the above equation is intractable; hence it is approximated using a two steps algorithm that finds first the minimal grounding of the network required to compute the probability. The second step returns the probability that F_1 holds using Gibbs sampling to sample the formula's groundings; then, the probability is the proportion of true groundings dividing by all samples.

This method is not applicable in the case of deterministic rules. To overcome this problem, [104] uses MC-SAT algorithm that combines MCMC and satisfiability verification. It is a sample-SAT algorithm inside an MC-MC process to uniformly sample from a set of satisfiable solutions. A lazy version of the algorithm is developed in lifted inference methods based on belief propagation and described in [107].

Since inference in the Markov logic network combines satisfiability verification which is NP-complete and MCMC which is #P-complete, it is intractable. In the next section, we present probabilistic soft logic, an SRL framework allowing polynomial inference.

2.4.2 Probabilistic Soft Logic

Probabilistic soft logic (PSL) is a SRL framework that defines a probabilistic distribution over a FOL. A PSL program associates non-negative weights to FOL rules and soft truth values in [0,1], making the inference in PSL a continuous optimisation problem.

To determine the satisfiability degree of rules, PSL uses the Lukasiewicz t-norm to relax the logic operators. The relaxation is presented in given by:

$$l_1 \tilde{\wedge} l_2 = max\{0, I(l_1) + I(l_2) - 1\}$$

$$l_1 \tilde{\vee} l_2 = min\{I(l_1) + I(l_2), 1\}$$

$$\tilde{\neg} l_1 = 1 - I(l1)$$

where the ˜represents the relaxation from the Boolean domain. A rule is defined as $r \equiv r_{head} \leftarrow r_{body}$ s.t r_{body} is a conjunction of literals and r_{head} is a single literal. An interpretation over a formula r defines whether the formula is satisfied and if it is not satisfied it defines the distance to satisfaction.

$$d_r(I) = max\{0, I(r_{body}) - I(r_{head})\}$$

Given a set of grounded atoms, a PSL program induces a distribution over possible interpretations I. Let R be the set of all grounded rules that are instances of a rule in the program and only mention atoms in l. The probability density function f over I is given by:

$$f(I) = \frac{1}{Z} exp[-\sum_{r \in R} \lambda_r (d_r(I)^p)]$$

with:

$$Z = \int_I exp[-\sum_{r \in R} \lambda_r (d_r(I)^p)]$$

where λ_r is the formula's wight, $d_r(I)$ is the distance to satisfiability, and $p \in \{1, 2\}$ provides the choice of two different loss functions. Depending on the choice of p, we convert the program after interpretation to linear optimization or a second-order cone program (SOCP).

In the next section, we present the inference and learning methods in PSL.

2.4.2.1 Inference

MPE inference aims to find the most common interpretation given evidence, which means maximizing the density function $f(I)$ that is equivalent to minimizing the summation in the exponent. Therefore, we search the truth assignment that maximize the function.

Initially, a truth value of 0 is assigned to the no evidence atoms; then the program is cast to SOCP problem, which can be solved in $O(n^{3.5})$ with n is the number of the relevant groundings i.e., rules with a no 0 distance to satisfaction. After that, we solve the SOCP by taking into consideration only the set of relevant rules. Once solving the SOCP, we update the set of relevant rules and resolve the SOCP problem; this process is repeated until no new relevant rules are found.

In the case of**MPA inference**, the estimation generally consists of estimating the volume of the slice of the convex polytope of non-zero density interpretations. In PSL, marginal distributions are estimated by collecting a histogram of sampled points following the hit-and-run MCMC scheme. Starting from a MAP state, the algorithm explores the convex polytope by first sampling a direction uniformly at random, followed by sampling a point on the line segment within the polytope.

2.4.2.2 Weights' learning

The rules' weights can be learned by maximizing the likelihood of the training data. The gradient of a weight λ_i is:

$$\frac{\partial}{\partial \lambda_i} log f(I) = -\sum_{r \in R_i} (d_r(I))^p + E[(\sum_{r \in R_i} (d_r(I))^p]$$

where R_i is the set of ground rules. Calculating the expectation is intractable. Hence, a common optimisation method approximates $d_r(I)$ with the most probable interpretation of I using MAP inference.

2.4.3 Limits

SRL methods constitute a robust framework that associates relational representation and probabilistic learning. However, despite the advances in research in this area, constructing powerful tools that ensure end-to-end reasoning is still challenging since all learning algorithms are supervised, which is not the case for all systems.

Part III

Contributions

Chapter 3

Quality of Sentiment Analysis Tools: The Reason of Inconsistency

Sentiment analysis has been introduced in critical life domains such as medicine, politics, and marketing, making consistency and accuracy of sentiment analysis tools a primary need. However, existing sentiment analysis tools present two types of inconsistency: intra-tool and inter-tool inconsistencies that are misleading and lead to poor business decisions.

To understand the reasons for these anomalies and the frequency of their presence in sentiment analysis tools, we present in this chapter the first extensive empirical study that quantifies and investigates the causes and factors impacting the inconsistencies.

The study demonstrates that sentiment analysis tools are not robust in the presence of adversarial examples, regardless of whether they are based on machine learning models, lexicons, or rules.

This chapter presents a formalization of the inconsistency problem on sentiment analysis tools, then presents the process we followed to create the benchmark we used in this study. After that, we present a statistical analysis followed by structural and semantic analysis to determine the relationship between documents, tools, and inconsistencies. Finally, we evaluate the inconsistencies based on hyperparameters of machine learning algorithms and present a set of recommendations to help with tools selection following the data type.

3.1 Problem Definition

Let D be a set of documents such that: $D = \{d_1, \ldots, d_n\}$, we consider the following definitions over D:

Definition 17. *Analogical Set*

It is a subset of D that contains semantically equivalent documents. Formally:

$$A_l \subseteq D, \text{ s.t: } \forall d_i, d_j \in A_l, d_i \overset{s}{\iff} d_j$$

where $\overset{s}{\iff}$ denotes semantic equivalence, and d_i and d_j are called analogical documents.

For instance, $A_1 = \{d_1, d_2\}$ in example 1.1 is an analogical set since $d_1 \overset{s}{\iff} d_2$ (they both express that the US is restricting Chinese technological investments).

Definition 18. *Sentiment Consistency*

For each dataset D and polarity functions set Γ, we define sentiment consistency as the two rules 1) and 2):

1) Intra-tool consistency. For each two documents d_i, d_j from the same analogical set A_l and for each polarity function $P_{t_k} \in \Gamma$, "intra-tool consistency" is defined as:

$$\forall d_i, d_j \in A_l \, \forall P_{t_k} \in \Gamma, \, P_{t_k}(d_i) = P_{t_k}(d_j) \tag{3.1}$$

This rule assumes that every two documents from A_l have the same polarity, which means that A_l has a unique polarity.

The intra-tool consistency consensus is adopted by many works in literature, such as [3], [29]–[33] that consider semantically equivalent documents with the same polarities.

We call semantically equivalent documents that violate the intra-tool consistency **adversarial examples**. Formally, d_i and d_j are adversarial examples if:

$$\exists d_i d_j \in A_l, P_* \in \Gamma \; s.t: \; P_*(d_i) \neq P_*(d_j) \tag{3.2}$$

For example, documents d_6 and d_7 in Table 1.1 violate intra-tool consistency so they are adversarial examples for the Senticnet sentiment analysis tool since $P_{Senticnet}(d_6) \neq P_{Senticnet}(d_7)$.

2) Inter-tool consistency. Inter-tool consistency assumes that the document has a unique polarity. It is defined as:

$$\forall d_i \in D \, \forall P_{t_k}, P_{t'_k} \in \Gamma \quad P_{t_k}(d_i) = P_{t'_k}(d_i) \tag{3.3}$$

In Table 1.1, the inter-tool consistency is not respected for the document d_3, since we have: $P_{rec\_nn}(d_3) \neq P_{Senticnet}(d_3)$. We call this case inter-tool inconsistency.

Definition 19. *Inconsistency classes.*

We distinguish three classes of inconsistencies based on the predicted polarity of documents:

1) Inconsistency of type Positive/Neutral

It represents the case where a tool P_{t_k} attributes Positive and Neutral polarities, respectively, to the semantically equivalent documents d_i and d_j:

$$d_i, d_j \in A_l \; P_{t_k} \in \Gamma, \; P_{t_k}(d_i) = \text{Positive and } P_{t_k}(d_j) = \text{Neutral} \tag{3.4}$$

or the case where the tools P_{t_k} and $P_{t'_k}$ attribute, respectively, Positive and Neutral polarities to the document d_i:

$$d_i \in D \; P_{t_k}, P_{t'_k} \in \Gamma \quad P_{t_k}(d_i) = \text{Positive and } P_{t'_k}(d_i) = \text{Neutral} \tag{3.5}$$

2) Inconsistency of type Negative/Neutral

It accrues when a tool P_{t_k} attributes Negative and Neutral polarities to the documents d_i and d_j that are semantically equivalent:

$$d_i, d_j \in A_l \; P_{t_k} \in \Gamma, \; P_{t_k}(d_i) = \text{Negative and } P_{t_k}(d_j) = \text{Neutral} \tag{3.6}$$

or when two tools P_{t_k} and $P_{t'_k}$ attribute the different polarities Negative and Neutral to the document d_i:

$$d_i \in D \; P_{t_k}, P_{t'_k} \in \Gamma \quad P_{t_k}(d_i) = \text{Negative and } P_{t'_k}(d_i) = \text{Neutral} \tag{3.7}$$

3) Inconsistency of type Negative/Positive

We got an inconsistency of type Negative/Positive when the tool P_{t_k} attributes, respectively, the polarities Positive and Negative to the semantically equivalent documents d_i and d_j

$$d_i, d_j \in A_l \; P_{t_k} \in \Gamma, \; P_{t_k}(d_i) = \text{Negative and } P_{t_k}(d_j) = \text{Positive} \tag{3.8}$$

or when the tools P_{t_k} and $P_{t'_k}$ attribute, respectively, the polarities Positive and Negative to the document d_i:

$$d_i \in D \; P_{t_k}, P_{t'_k} \in \Gamma \quad P_{t_k}(d_i) = \text{Negative and } P_{t'_k}(d_i) = \text{Positive} \tag{3.9}$$

3.2 Benchmark Construction

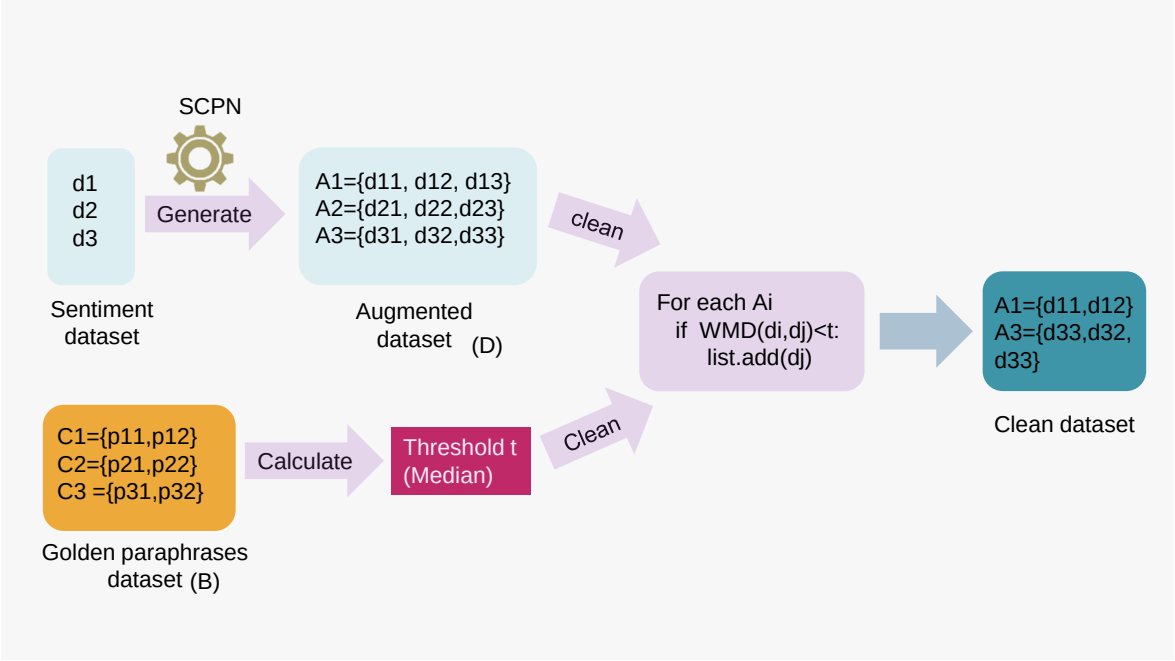


Figure 3.1: Benchmark construction procedure

To evaluate the tools' inconsistencies and accuracy, a dataset of analogical sets with polarity labels is needed. Unfortunately, to the best of our knowledge, this dataset does not exist publicly. Therefore, we built a benchmark of analogical sets labeled with polarities based on publicly available sentiment datasets that we augmented with paraphrases.

This method allows generating analogical datasets with a large number of documents, instead of the straight-forward method which consists of labeling paraphrases datasets manually.

Therefore, we have extended five publicly available datasets using the syntactically controlled paraphrase networks (SCPNs) [1]. We have chosen SCPN because it is unsupervised and generates large analogical sets. Figure 3.1 illustrates the process we followed to construct the benchmark.

First, we have augmented the sentiment datasets using SCPN, which generates a dataset of analogical sets. After that, we have calculated the WMD between every two paraphrases in the golden dataset. Then, we have calculated the threshold distance in that dataset and clean the generated dataset based on this distance.

Next, we present a detailed explanation of the benchmark construction process.

3.2.1 Base Datasets

To construct the paraphrases benchmark annotated with polarity labels, we select five publicly datasets based on their popularity [10], [44]–[50], size, having at least three polarity labels, and containing different types of documents which allows evaluating inconsistency based on the document type:

- The *News Headlines* [108] dataset contains short sentences related to financial news headlines collected from tweets and news RSS feeds and annotated by experts with polarity values $\Omega \in [-1, 1]$.

- The *Stanford Sentiment Treebank (SST)* [40] is built based on movie reviews and annotated using Mechanical Turk crowdsourcing platform [1] with value $\Omega \in [0, 1]$.

- The *Amazon Reviews Dataset* [109] contains product reviews from the Amazon websites, including text reviews and the product's five-level rating as a polarity score.

- The *US airlines tweets dataset*[2] that contains tweets about the US airline companies and is annotated based on the emojis present in the tweets.

- The *first GOP debate dataset* [3] that contains tweets related to the debate between Trump and Clinton.

We refer to these datasets, respectively, as **News**, **SST**, **Amazon**, **US airlines**, **FGD** datasets. Statistics about these datasets are presented in Table 3.2.

We also consider two other datasets, *Microsoft research paraphrases corpus* [110] and STS [111], that contains valid paraphrases:

- *Microsoft research paraphrases corpus (MSR)* [110] contains paraphrases pairs collected from multiple news sources on the web annotated by experts with binary labels: 1 in case the sentence pairs are paraphrases 0 otherwise.

- *Semantic Text Similarity data-set (STS) [112]* is a dataset used for semantic similarity evaluation tasks. It contains sentence pairs collected from news websites and forums annotated using crowd-sourcing platforms. Each pair of sentences has a similarity score in [0,5].

Statistics about these datasets are presented in Table 3.1. We refer to the two datasets as MSR and STS.

[1] https://www.mturk.com/
[2] https://www.kaggle.com/crowdflower/twitter-airline-sentiment
[3] https://www.kaggle.com/crowdflower/first-gop-debate-twitter-sentiment

Statistics	MSR	STS
Nb pairs	5801	8628
Source	News	Forum, News, Captions
Annotation method	Expert annotation	Crowd -sourcing

Table 3.1: Statistics of paraphrases datasets.

Datasets	#Negative	#Positive	#Neutral	#Total
Amazon	847	9452	701	11000
FGD	8493	2236	3142	13871
News	451	653	38	1142
SST	863	872	372	2107
US airlines	9178	2363	3099	14640
Total_data	19832	15576	7352	42760

Table 3.2: Statistics of base datasets

Datasets	Original Size	Nb Clusters	Size After Augmentation
Amazon	11000	11000	147896
FGD	13871	13871	141200
News	1142	1142	10937
SST	2107	2107	12445
US airlines	14640	14640	145900

Table 3.3: Statistics on augmented datasets

3.2.2 Augmented Datasets

We extend the previously mentioned sentiment labeled datasets using the method described in [1], that automatically generates syntactically controlled paraphrases for each document (between 2 and 10 documents) that should have the same polarity. The statistics of the augmented datasets are presented in Table 3.3.

3.2.3 Refined Datasets

The problem with the tool in [1] is that it produces wrong predictions for 20% of the cases, which may affect the quality of the analysis. To resolve this problem and to improve the quality of the benchmark by reducing the produced error rate, we propose a protocol that minimizes the human effort and the cost of data quality verification. The two-steps protocol is described in Algorithm 5 to obtain clean dataset and Algorithm 6 to calculate the similarity threshold needed in step 2 of the protocol.

We consider two analogical datasets B, and D, where B is the valid set of paraphrases from the MSR [113] and D is generated automatically. We calculate the Word Mover Distance (WMD) [114] between each pair of paraphrases in B. For that, we consider the word2vec embedding matrix $X \in R^{m \times n}$ such that m is the size of the word embedding vector [70]. The distance between two words is the euclidean distance called distance transportation cost (c). We also define the transformation matrix $T \in R^{n \times n}$ which represents the proportion of each word in the transformation of document d to the document d'. The weights of this matrix must verify the conditions: $\sum_{i=0}^{n} T_{ij} = d_i$ and $\sum_{j=0}^{n} T_{ij} = d'_j$. Then, WMD is the solution of the objective function:

$$min \sum_{i=0}^{n} T_{ij} c(i,j)$$
$$s.t \sum_{i=0}^{n} T_{ij} = d_i \; and \sum_{j=0}^{n} T_{ij} = d'_j$$

Once the WMD has been calculated between all couples of paraphrases in B, we calculate the population's median value that we consider as the threshold distance. The main steps of this part of the method are described briefly in Algorithm 6.

Algorithm 5 Clean_dataset

Input: $D = \{A_1 = [S_{11}, \ldots, S_{1m}], \ldots, A_n = [S_{n1}, \ldots, S_{nm'}]\}$
B: Golden dataset
Output: Clean dataset

```
//Step1: Get the distance threshold
t = Calculate_threshold(B)
//Step2: Verify if generated elements respect the max threshold t
for each A_i ∈ B :
    S_o = A_i[0]
        for each S_ik ∈ A_i :
            if  WMD(S_ij, S_o) < t :
                NewA.add(S_ij)
    list.add(NewA)
return list
```

Algorithm 6 Calculate_threshold

Input: Golden dataset $B = \{A_1 = [S_{11}, S_{12}], \ldots, A_n = [S_{n1}, S_{n2}\}]$
Output: Similarity threshold t

1: **for each** $A_i \in B$:
2: //**Step1:***calculate WMD distance*
3: $dist = WMD\ (A_i[0], A_i[1])$
4: *list.add(dist)*
5: //**Step2:** *Calculate the threshold distance*
6: *t= median(list)*
7: *return t*

After defining the max distance threshold t, we calculate the WMD distance between each two documents S_{ij} and S_{ik} in the analogical sets A_i of D. Then, we only keep the generated documents with a distance lower than t. The output after this step is a refined dataset of analogical sets containing semantically close documents.

In the experiments, we only use the refined datasets, while we use the base datasets for training machine learning models.

For simplicity, we refer to generated datasets using the same name as the base datasets. For example, SST refers to the extended, then refined version of the dataset SST. The statistics of our benchmark are displayed in Table 3.4.

Datasets	# Elements	# Clusters	# Positive	# Neutral	# Negative
Amazon	21483	10704	16944	2028	2511
FGD	28192	9298	4319	6296	1384
News	1580	831	890	36	654
SST	3504	1033	1492	628	1384
US airlines	38236	12894	5107	6816	26313
Total	92995	34760	28752	15804	48439

Table 3.4: Statistics of clean benchmark

3.3 Experiments

In this section, we conduct a thorough experimental evaluation of the robustness of existing sentiment analysis tools in the presence of adversarial examples.

First, we introduce our experimental setup; then, we evaluate the different techniques based on a statistical analysis followed by an in-depth structural and semantic

analysis. For the sake of reproducibility, we make publicly available our datasets and codebases[4].

All our experiments are implemented in python and java, and were conducted on a server with 75GB of RAM and two Intel Xeon E5-2650 v4 2.2GHz CPUs.

3.3.1 Experimental Setup

3.3.1.1 Tools

In our experiments, we use the sentiment analysis tools summarized in Table 3.5. We use SentiWordnet [41] as a representative method for the lexicon-based category because of its popularity and coverage.

We also use two representative methods for rule-based approaches: Vader [8], whose high quality has been verified by human experts, and SenticNet [29], which is a lexicon of concepts. Using these different lexicons allows us to evaluate inconsistencies on two different levels: concepts and word level.

In the learning-based methods, we use three machine learning tools: RecNN [40] that learns word embeddings, Text_ CNN [44] that uses a pre-trained word embedding method, and Char_CNN [60], that uses two levels of embedding.

All these methods are powerful, widely used, and use different embedding types, which allows us to study the relationship between the embedding type and inconsistencies.

We note that we implemented the two methods [60] and [44], and retrained the networks on the used datasets which produce a total of nine models (a model for each embeeding type and dataset).

We refer to sentiment analysis tools by their polarity functions (see definition 3), i.e., SenticNet as $P_{senticnet}$, Sentiwordnet as, $P_{sentiwordnet}$, Vader as P_{vader}, Text_CNN as P_{text_cnn}, Char_CNN as P_{char_cnn}, and RecNN as P_{rec_nn}.

3.3.1.2 Metrics

In this part, we present the different metrics used in our experiments besides WMD described previously:

– We evaluate **Accuracy** by calculating the rate of correctly predicted polarities compared to the golden polarity (human annotation) as follows:

$$Accuracy = \frac{\sum_{i=0}^{n} \mathbb{1}_{(P_h(d_i)=P_*(d_i))}}{n}$$

[4]https://github.com/AbCd256/Quality-of-sentiment-analysis-tools

Technique	Method	Mean Accuracy	Test Dataset
Lexicon based method	Sentiwordnet [41]	Binary: 51% [115]	MPQA dataset [116]
Rule based methods	SenticNet [29]	Binary: 93.7%	Blitzer [117] Movie reviews [118]
	Vader [8]	Fine grained:79%	Amazon Reviews Movie Reviews Tweets NY times Editorials
Machine learning (RNN)	RecNN[40]	Fine grained : 45% Binary: 85.4%	Sentiment tree bank[40]
Machine learning (CNN)	CNN[44]	Fine grained: 48% binary: 84.5%	Movie Reviews[118] sentiment tree bank [40] Costumer reviews [119]
	CharCNN[60]	Fine graind: 48.7% Binary : 85.8%	Sentiment treebank [40] stanford twitter Sentiment [120]

Table 3.5: Sentiment analysis tools used in the evaluation

– The **cos similarity** between two vectors V and U is given by:

$$Cos\_sim(V,U) = \frac{U.V}{\|V\|.\|U\|}$$

– We evaluate the **intra-tool inconsistency rate** for each document d_j in the analogical set A and sentiment analysis tool t_k as the proportion of documents d_z with a different polarity than $P_{t_k}(d_i)$. We write

$$\begin{aligned} &\forall d_i \in A\,, P_{t_k} \in \Gamma\,, inc_{in}(d_i, P_{t_k}) = \frac{card(S)}{n-1} \\ &\text{s.t } S = \{d_j \in A | P_{t_k}(d_i) \neq P_{t_k}(d_j)\} \end{aligned} \tag{3.10}$$

The intra-tool inconsistency rate of an analogical set A is the mean of different intra-tool inconsistency rates of its documents:

$$inc_{in}(A, P_{t_k}) = \frac{\sum_{j=1}^{n} inc_{in}(d_j, P_{t_k})}{n},\ n = card(A) \tag{3.11}$$

– We measure the **inter-tool inconsistency rate** for each tool t_k and document d_j as the rate of tools $t_{k'}$ that give different polarities to the document d_j than P_{t_k}. We write:

$$\begin{aligned} &\forall P_{t_k} \in \Gamma, \forall d_i \in A,\ inc_{inter}(d_i, P_{t_k}) = \frac{card(S')}{m-1} \\ &\text{s.t } S' = \{P_{t'_k} \in \Gamma | P_{t'_k}(d_j) \neq P_{t_k}(d_j)\} \end{aligned} \tag{3.12}$$

The inter-tool inconsistency rate in the set Γ is the mean of inconsistency rates

of the different tools:

$$inc_{inter}(d_j, \Gamma) = \frac{\sum_{k=1}^{m} inc_{inter}(d_j, P_{t_k})}{m},\ m = card(\Gamma) \tag{3.13}$$

3.3.2 Statistical Analysis

In this sub-section, we evaluate the quality of sentiment analysis tools statistically. The goal of this evaluation is to answer the following questions:

(1) Is intra-tool inconsistency a rare event?

(2) What types of inconsistencies exist?

(3) Are there domains that are more prone to inconsistencies?

3.3.2.1 Intra-tool Inconsistency Rate

The purpose of this experiment is to evaluate the intra-tool inconsistencies that occur in different sentiment analysis tools. We first check the number of intra-tool inconsistencies on a golden dataset of valid paraphrases. Then, we evaluate inconsistencies in the generated benchmark:

Intra-tool Inconsistency Rate on Golden Dataset

This experiment evaluates the intra-tool inconsistencies in the two paraphrases datasets of Table 3.1.

We analyze the pairs of paraphrases using the sentiment analysis tools P_{text_cnn}, P_{rec_nn}, $P_{senticnet}$, and $P_{sentiwordnet}$; then, we count the inconsistent pairs (pairs of paraphrases with different polarities). The results are plotted in Figure 3.2.

We observe an important number of inconsistencies on the two datasets in all tools (a proportion of 23%, 22%, 30%, and 25% of inconsistent pairs within the tools P_{text_cnn}, P_{rec_nn}, P_{sentic}, and $P_{sentiwordnet}$ respectively) .

We notice that intra-tool inconsistencies are not rare, and they are commonly present in sentiment analysis tools. Next, we analyze inconsistencies on a larger dataset, using more sentiment analysis tools from different categories to generalize the findings.

Intra-tool Inconsistency Rate on the Generated Benchmark

In this experiment, we evaluate the intra-tool inconsistencies in each analogical set using Equation (3.11), then we calculate the mean inconsistency of tools in each dataset. We display the results in Figure 3.3.

Sub-figures 1-6 represent the tools' mean inconsistency on datasets, while sub-figure 7 shows the proportion of analogical set with an intra-tool inconsistency different from 0. We notice that 44% of analogical sets have an intra-tool inconsistency different

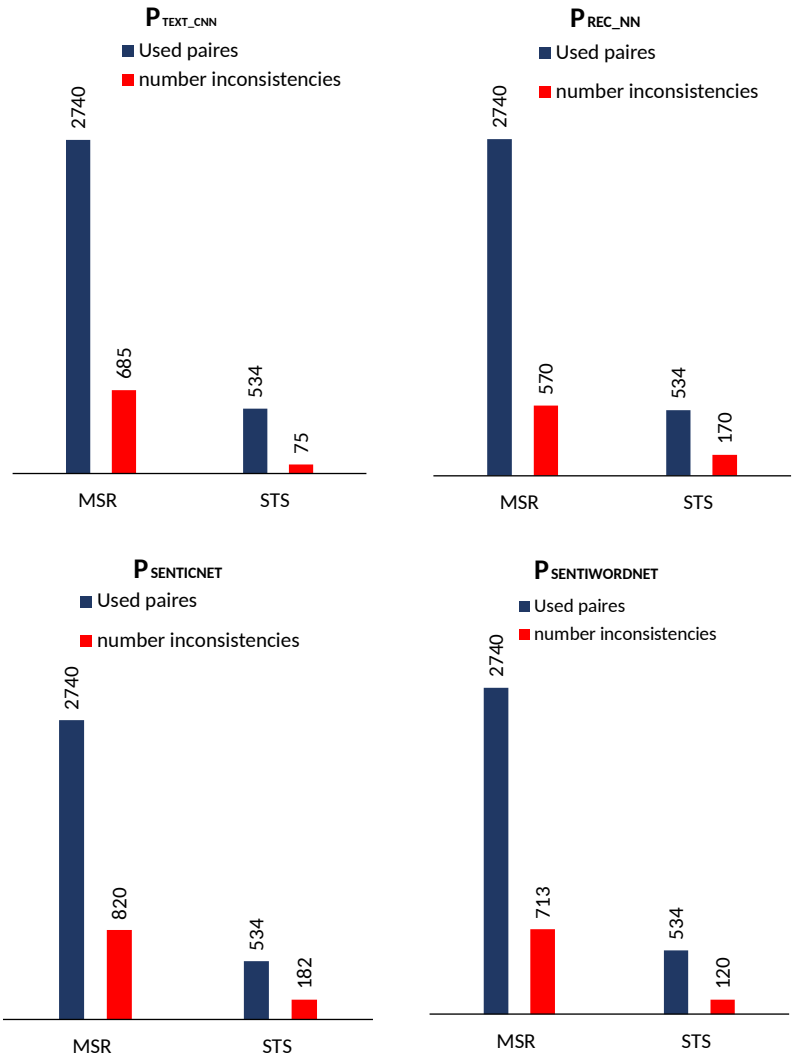


Figure 3.2: Intra-tool inconsistencies on golden paraphrases datasets MSR and SST with the tools P_{Text_cnn}, P_{rec_nn}, $P_{senticnet}$, and $P_{sentiwordnet}$

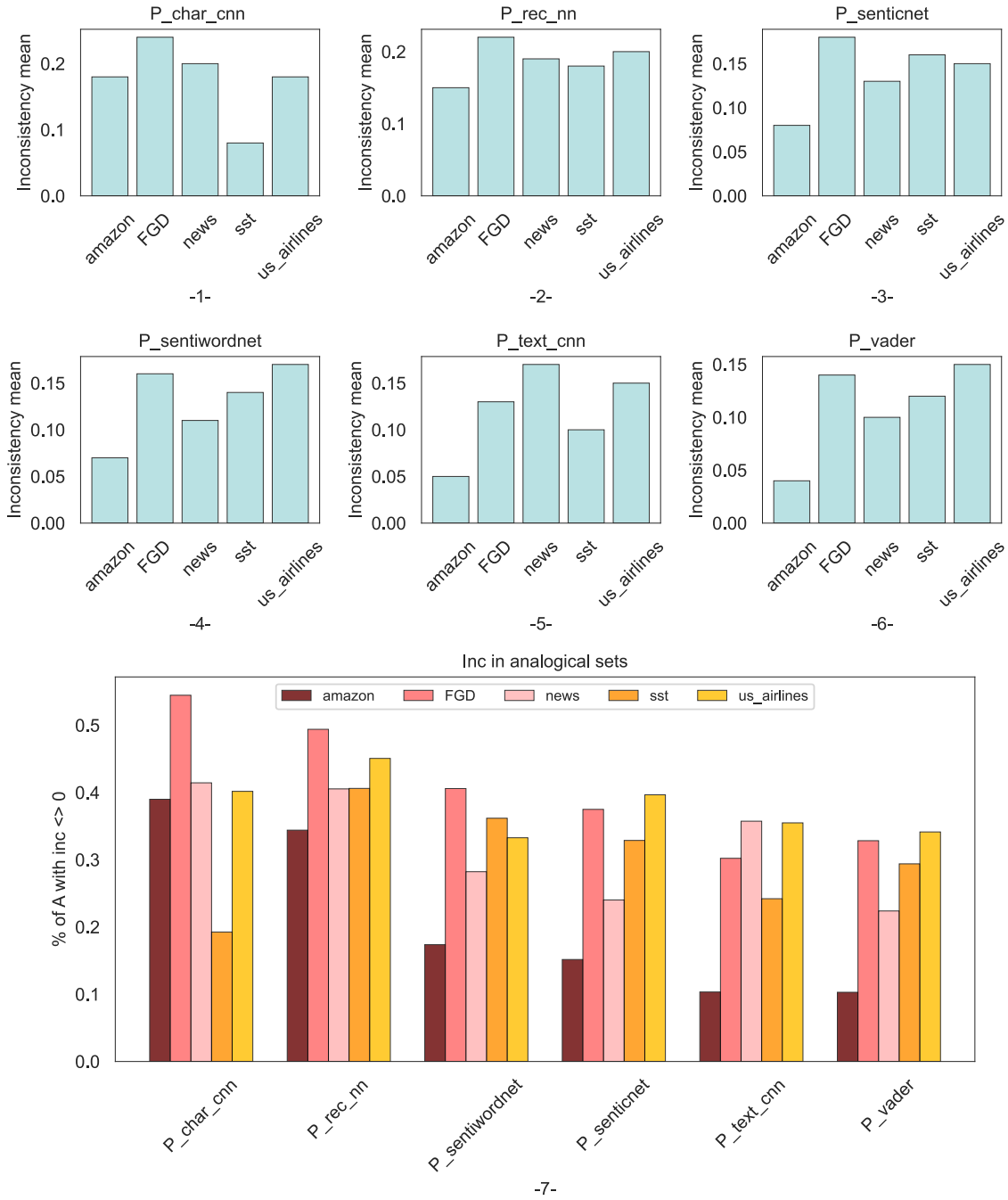


Figure 3.3: Intra-tool inconsistency distribution and the ratio of tools with inconsistency different from 0 of tools in Table 3.5

from 0 on P_{char_cnn}, 40% on P_{text_cnn}, 26.6% on $P_{sentiwordnet}$, 33.6% on $P_{senticnet}$, 44.8% on P_{rec_nn} and 8.5% on P_{vader}.

We notice a high intra-tool inconsistency degree on the machine learning-based methods P_{char_cnn} and P_{rec_nn} (an average of 0.17 and 0.188 respectively) and a lower inconsistency degree of 0.11 on P_{text_cnn}. We notice also lower inconsistency degrees on lexicon and rule-based methods ($P_{sentiwordnet}$ 0.104, and P_{vader} 0.11).

Therefore, we can deduce that lexicon-based methods are more consistent than machine learning methods.

We also notice that tools are more inconsistent on the *FGD* and *US airlines* datasets. We notice also high inconsistency on *SST* even for the P_{rec_nn} that was trained on *SST* dataset.

Summary

Intra-tool inconsistencies are frequent on different sentiment analysis tools, which motivate a more in-depth study to cover the causes of such anomalies in the next sections (Sections 3.3.3 and 3.3.4).

3.3.2.2 Inter-tool Inconsistency Rate

In this experiment, we calculate the inter-tool inconsistency degree according to Equation (3.12), then we calculate the mean inter-tool inconsistency of tools on the datasets.

Sub-figures 1-6 of Figure 3.4 report the mean inter-tool inconsistency between tools on our datasets, and Sub-figure 7 shows the percentage of documents where tools have an inconsistency degree of 1.

We notice that there is a large degree of inter-tool inconsistency between different sentiment analysis tools. For example, the mean inter-tool inconsistency is 0.535 on P_{char_cnn}, 0.52 on P_{rec_nn}, 0.575 on $P_{senticnet}$, 0.547 on $P_{sentiwordnet}$, and 0.53 on P_{text_cnn}.

We notice also that P_{char_cnn} contradict all tools (inconsistency degree =1) on 12.35% of cases, $P_{sentiwordnet}$ on 11.7%, P_{text_cnn} on 18.2% and P_{vader} on 13.7%.

The highest inter-tool inconsistencies are on the *News* dataset with a mean inconsistency degree of 0.62, which can be explained by the challenging nature of the documents present in this dataset which are short in length, factual, and meaningful. Therefore, it is difficult to catch the polarity features from the sentences.

Summary

These experiments show that inter-tool inconsistency is a frequent event that occurs with a high degree on the Financial *News* dataset that represents relevant data for critical decision-making processes.

This motivates us to refine our analysis to unveil and explain the causes behind

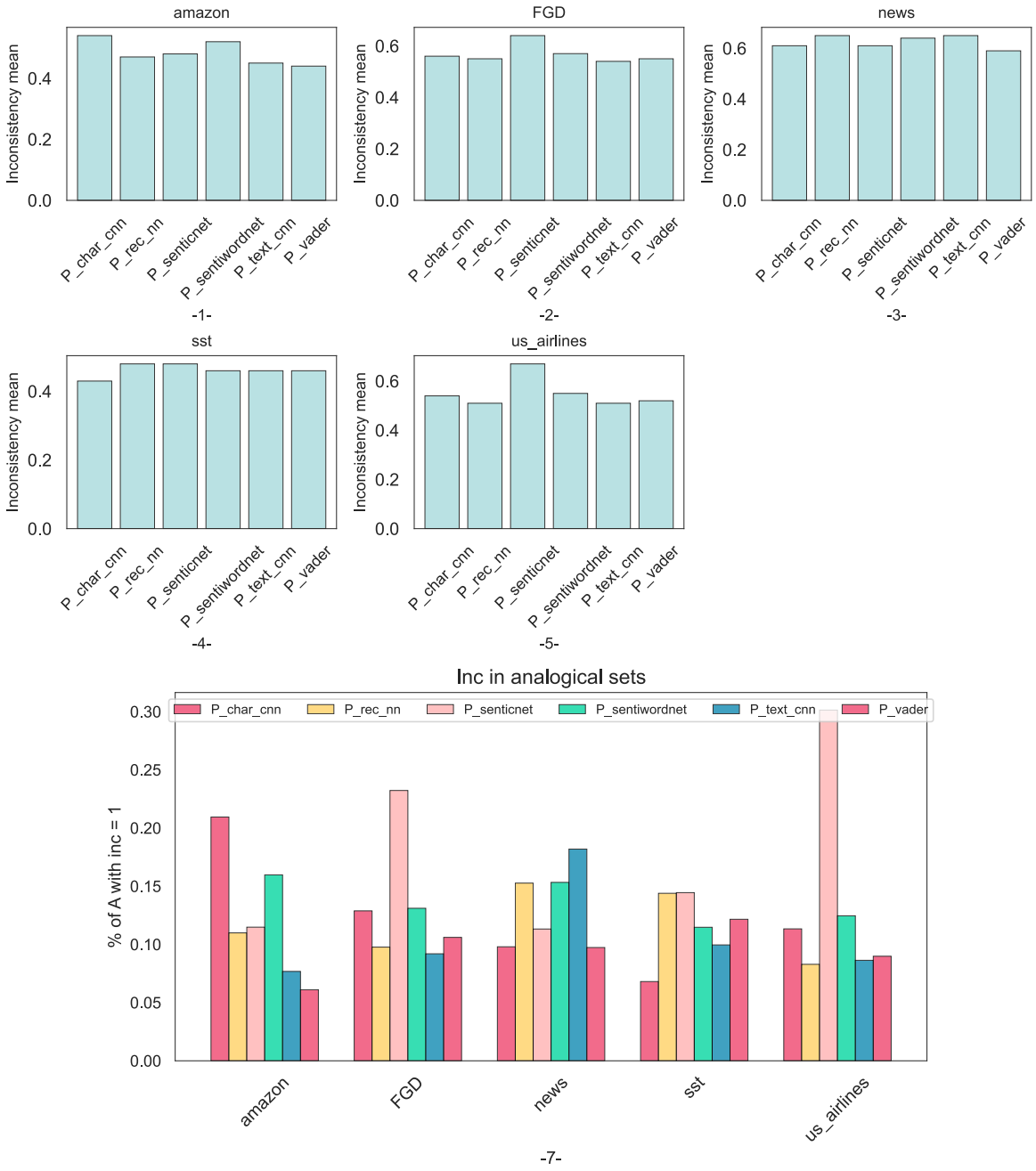


Figure 3.4: Inter-tool inconsistency distribution and the ratio of documents with inter-tool inconsistency equal to 1

these inconsistencies in the next sections (Sections 3.3.2.4 and 3.3.2.5).

3.3.2.3 Intra-tool Inconsistency Type

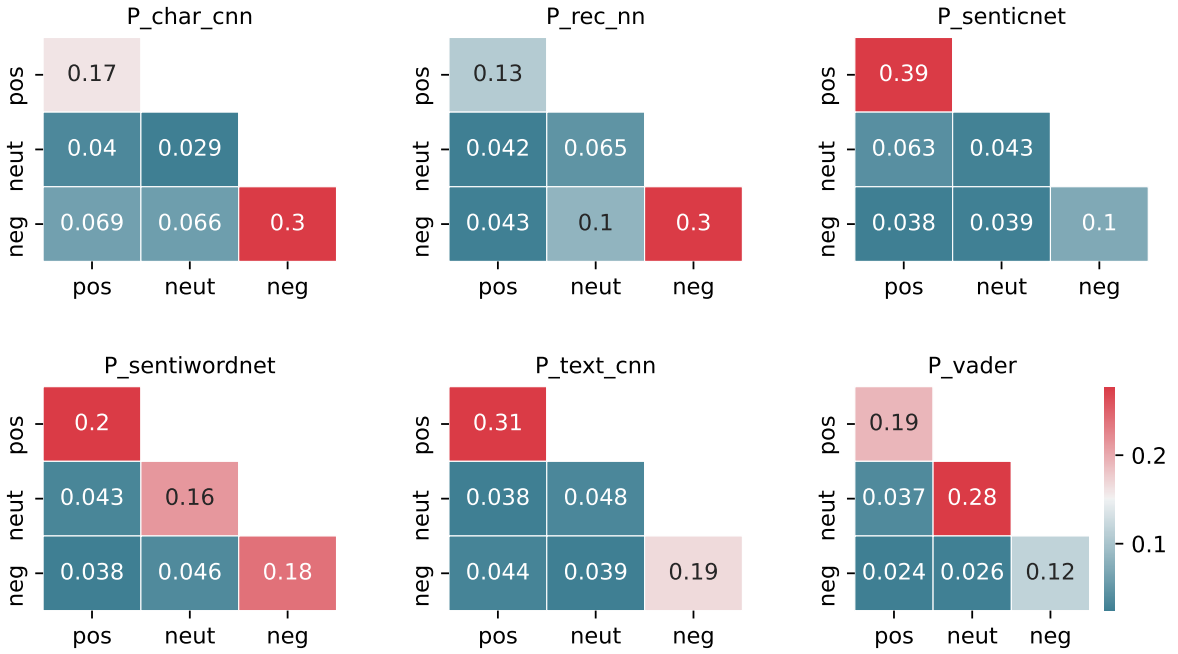


Figure 3.5: intra-tool inconsistency type

In this experiment, we calculate the mean of polarity inconsistencies by classes (see definition 19). We consider the document consistency in this case as a $M \in R^{3\times3}$ matrix where M_{ij} is the proportion of equivalent documents that got respectively the polarities i and j by the sentiment analysis tool k formally: $M_{ij} = \frac{card(S)}{n}$ such that for a document d_z, $S = \{d_{z'} \in A | P_k(d_z) = i \wedge P_k(d_{z'}) = j\}$. The consistency matrix of the analogical set A_i is the average of documents consistency matrices in A_i, and the inconsistency matrix of the dataset D is the average of inconsistency matrices of all analogical sets in D.

In Figure 3.5, we display the intra-tool inconsistency matrices for each tool and dataset (inconsistencies are the no diagonal values).

This experiment demonstrates that the inconsistencies are not only ones of type (Neutral/Positive)[5], or (Neutral/Negative).

In fact, an important proportion of inconsistencies are of type (Positive/Negative). For instance, most inconsistencies on P_{char_cnn} and P_{char_cnn} are of type Positive/Negative, while on $P_{senticnet}$ the inconsistency degree of Positive/Negative inconsistencies is 0.038, 0.043 on P_{rec_nn}, and 0.038 on $P_{sentiwordnet}$.

[5]The notation "(Neutral/Positive)" refers to two analogical documents with polarities Neutral and Positive, respectively.

We observe that the inconsistencies of type (Negative/Positive) are frequent in learning-based tools compared to lexicon-based tools. We also note that lexicon-based methods predict the polarity of an important proportion of documents as Neutral: 0.28 in P_{vader} and 0.16 in $P_{sentiwordnet}$ compared to other tools because those methods require the presence of a polar word in the document to classify it as Positive or Negative, which is not the case of real data.

Summary

This experiment shows that we have several types of intra-tool inconsistencies including Negative/Positive.

3.3.2.4 Inter-tool Inconsistency Type

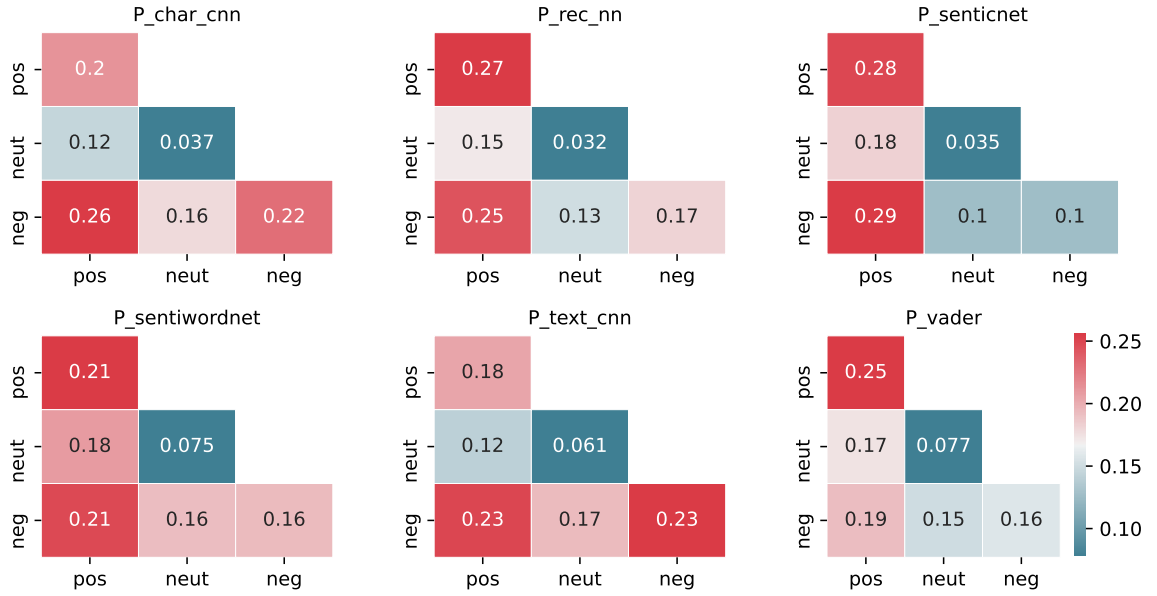


Figure 3.6: Inter-tool inconsistency type

In this experiment, we categorize the prediction results of the sentiment tools on documents according to polarity following definition 19. The inter-tool consistency is represented with a matrix $M \in R^{3\times3}$ where M_{ij} is the proportion of tools that attribute respectively the polarities i and j to the document d formally: $M_{ij} = \frac{card(S)}{m}$ with m the size of Γ such that for a document $d_z \in A$, a polarity function $P_{t_k} \in \Gamma$, $S = \{P_{t_{k'}} \in \Gamma | P_{t_k}(d_z) = i \wedge P_{t'_k}(d_z) = j\}$.

The experiment results are displayed in Figure 3.6. We observe that most inconsistencies are of (Positive/Negative) between all tools.

3.3.2.5 Discussion

In the above experiments, we have demonstrated that intra-tool inconsistency is a frequent anomaly in sentiment analysis tools, particularly for machine learning-based

Id	Document	Score	Polarity
43060	' (The Cockettes) provides a window into a subculture hell-bent on expressing itself in every way imaginable. '	0.625	Positive
5	' (the cockettes) provides a window into a subculture hell-bent on expressing itself in every way imaginable	0.375	Negative
179516	's a conundrum not worth solving	0.38889	Negative
179517	's a conundrum not worth solving.	0.5	Neutral

Table 3.6: Example of inconsistencies on the *SST* dataset

methods.

After refining our analysis by categorizing inconsistencies by type, we found less intra-tool inconsistencies on methods that use a word dictionary. These methods require the presence of a polar word in the document to classify it as positive or negative. Otherwise, they consider the review as neutral, contrary to machine learning and lexicon-based methods with a concept dictionary.

We notice most intra-tool inconsistencies on the *FGD*, *US airlines*, and *SST* datasets, and the most inter-tool inconsistencies on the *News* datasets. We explain this by the presence of abbreviations and syntax errors on the *FGD* and *US airlines* datasets.

We analyzed the datasets *News* and *SST* to unveil the causes of inconsistencies, and we found that in the *News* dataset, the documents are meaningful and short in length. Hence, it is difficult for tools to extract the sentiment features present in the review. While on *SST*, we notice the presence of semantically equivalent items with different polarities. Table 3.6 represents a snapshot of inconsistencies on *SST*

We see that the documents with identifiers 43060 and 5 express the same information.

However, there are missing punctuation (. and ') that do not affect the document's polarity, and the movies' title (The Cockettes) is written without any uppercase letters, which generally does not affect the polarity.

Certainly, uppercase words may embed sentiment as it is mentioned in [8], but it impacts the strength of the sentiment and does not switch the polarity from Positive to Negative.

3.3.3 Structural Analysis

In this part, we study the impact of analogical set structures on the intra-tool inconsistency by checking if inconsistencies are due to the semantic or syntactic distance between documents. In other words, checking whether the inconsistencies depend on the cos_cim as a measure to capture the syntax difference between sentences or on the WMD_Sim as a measure to capture their semantic.

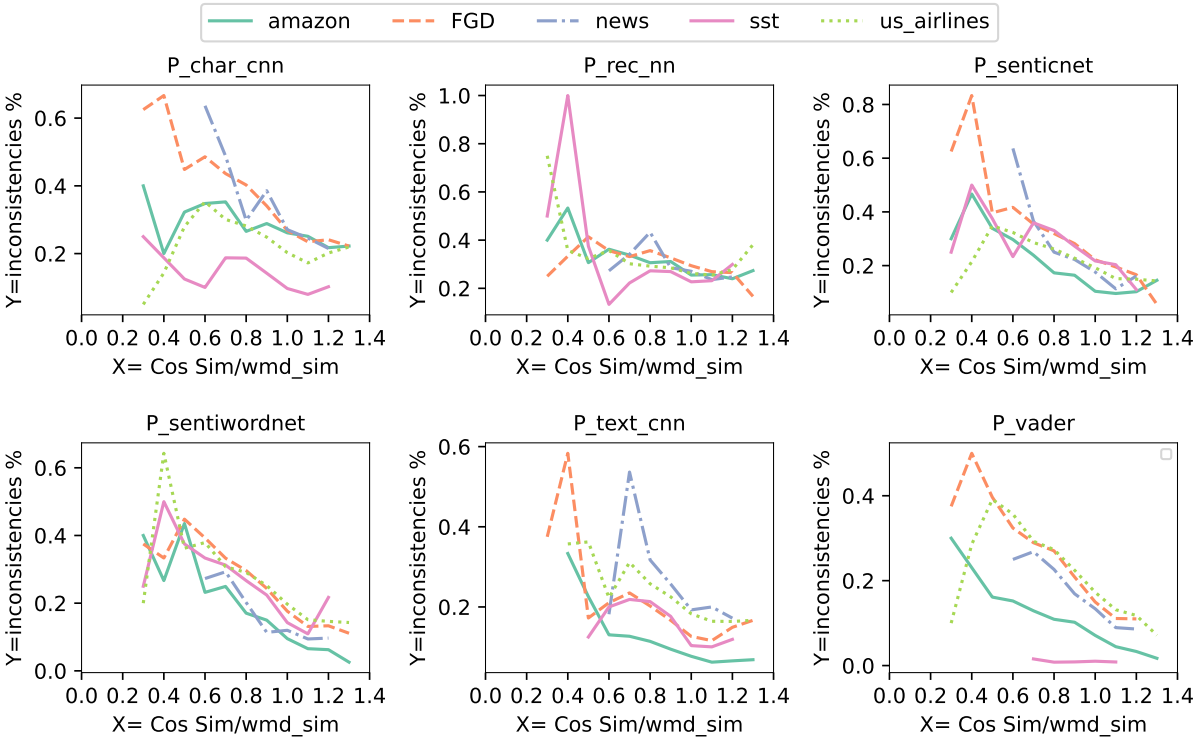


Figure 3.7: Inconsistency degree and similarity

We calculate the cos similarity between the bag-of-words representations of the documents. The Cos similarity measures the syntax similarity between the documents, since we considered the synonyms and abbreviations as different words. The WMD based similarity is calculated between the Word2Vec representations of the words in the documents, which allows to handle semantics.

In this experiment, we verify the relation between the syntactic difference measured by cos_cim, the semantic difference measured by WMD_sim, and the inconsistencies between every two analogical documents $d_i, d_j \in A$.

We first represent the documents as a bag-of-words such that $d_i = [v_1, \ldots, v_n]$, with n being the vocabulary size and v_w the occurrence of the word w in the document d_i. Then, we calculate $cos\_sim = (d_i, d_j)$ between each two documents d_i and d_j of A.

We calculate also the WMD between the word2vec representation of the two documents d_i, $d_j \in A$ and we convert it to a similarity score by using $WMD\_Sim = \frac{1}{1+WMD(d_i,d_j)}$.

After that, we calculate the ratio $\delta = \frac{cos\_sim(d_i,d_j)}{WMD\_sim(d_i,d_j}$ which is high for large values of cos_sim and low values of WMD_sim, and low for large values of WMD_sim and low values of cos_sim.

The percentage of inconsistent documents is calculated as: $\forall d_i, d_j \in A$, $P_{t_k} \in \Gamma\ \mathbb{1}_{(P_{t_k}(d_i) \neq Pt_k(d_j))}$. We categorize the results by δ values (X-axis) and calculate the proportion of inconsistencies (adversarial examples) to the total number of elements

that have δ (Y-axis). The results are displayed in Figure 3.7.

We notice that we have more inconsistencies for a low δ: a high WMD_sim and low Cos_sim on all tools for all datasets comparing to the inconsistencies that we have for a high δ i.e., a low WMD_sim and high Cos_sim, which indicates that sentiment analysis tools are much influenced by the syntactic variation of the sentence.

Summary

We observed more inconsistencies for high values of WMD_sim and low values of Cos_Sim on all tools.

3.3.3.1 Discussion

In the above experiment, we have evaluated the relationship between analogical set structure, document structure, and inconsistency.

The results show that most tools have a large proportion of inconsistencies between documents with a high semantic similarity degree and a low syntactic similarity degree. We learn from these experiments that the structure of the sentence affects the tool vulnerability for inconsistencies.

According to [5], the document structure may embed a sentiment, but not each modification in the structure implies a sentiment variation as shown in the examples of Table 3.6. It is crucial to consider this point when developing tools since most sentiment analysis applications are in uncontrolled environments like social media and review websites. In an uncontrolled environment, there are many unintentional documents restructuring that do not imply a polarity switching.

3.3.4 Semantic Analysis

The semantic analysis aims to verify the relationship between the document subjectivity and inconsistencies by answering the following questions:

1. Are inconsistencies frequent between polar facts or opinionated documents?
2. Do fewer inconsistencies imply higher accuracy? In other words, we evaluate the efficiency of tools in terms of accuracy and inconsistency.

3.3.4.1 Inconsistencies and Polar Facts

In this experiment, we study whether inconsistencies occur only between polar facts (i.e., the documents that do not include a polar word, such as: "this product does its job"), or even between opinionated documents (i.e., documents that contain polar words such as: "yeah! this product does its job").

Tools	*Amazon*			*News*			*SST*			*FGD*			*Usairlines*			Mean		
	Acc	Intra-tool inc	Inter-tool inc	Acc	Intra-tool inc	inter-tool inc	Acc	intra-tool inc	Inter-tool inc	Acc	Intra-tool inc	Inter-tool inc	Acc	Intra-tool inc	Inter-tool inc	Mean Acc	Mean Intra-tool inc	Mean Inter-tool inc
P_{char_cnn}	46.6%	0.18	0.54	47.9%	0.197	0.608	76.14%	0.078	0.43	47.4%	0.24	0.56	62.4%	0.175	0.538	56.1 %	0.17	0.535
P_{rec_nn}	52.5%	0.155	0.45	40.52%	0.188	0.649	65.08%	0.175	0.46	54.6%	0.22	0.538	64.4%	0.2	0.513	55.42%	0.188	0.522
$P_{sentic net}$	59.5%	0.077	0.48	52.67%	0.132	0.607	45.84%	0.164	0.48	31.15%	0.18	0.64	24.9%	0.15	0.67	42.8%	0.14	0.575
$P_{sentiwordnet}$	41.98%	0.07	0.52	33.01%	0.102	0.636	48.64%	0.137	0.46	41.69%	0.159	0.57	45.85%	0.17	0.55	42.23%	0.104	0.547
P_{text_cnn}	84.02%	0.03	0.47	49.04%	0.167	0.647	46.3%	0.09	0.48	70.01%	0.13	0.547	64.11%	0.15	0.51	62.696%	0.113	0.53
P_{vader}	56.5%	0.04	0.44	45.5%	0.1	0.6	51.4%	0.124	0.46	43.8%	0.14	0.55	49.9%	0.15	0.52	49.42%	0.109	0.514
MV	67.22%			**56.3%**			61.4%			53.8%			60%			N/A		

Table 3.7: Accuracy and inconsistency

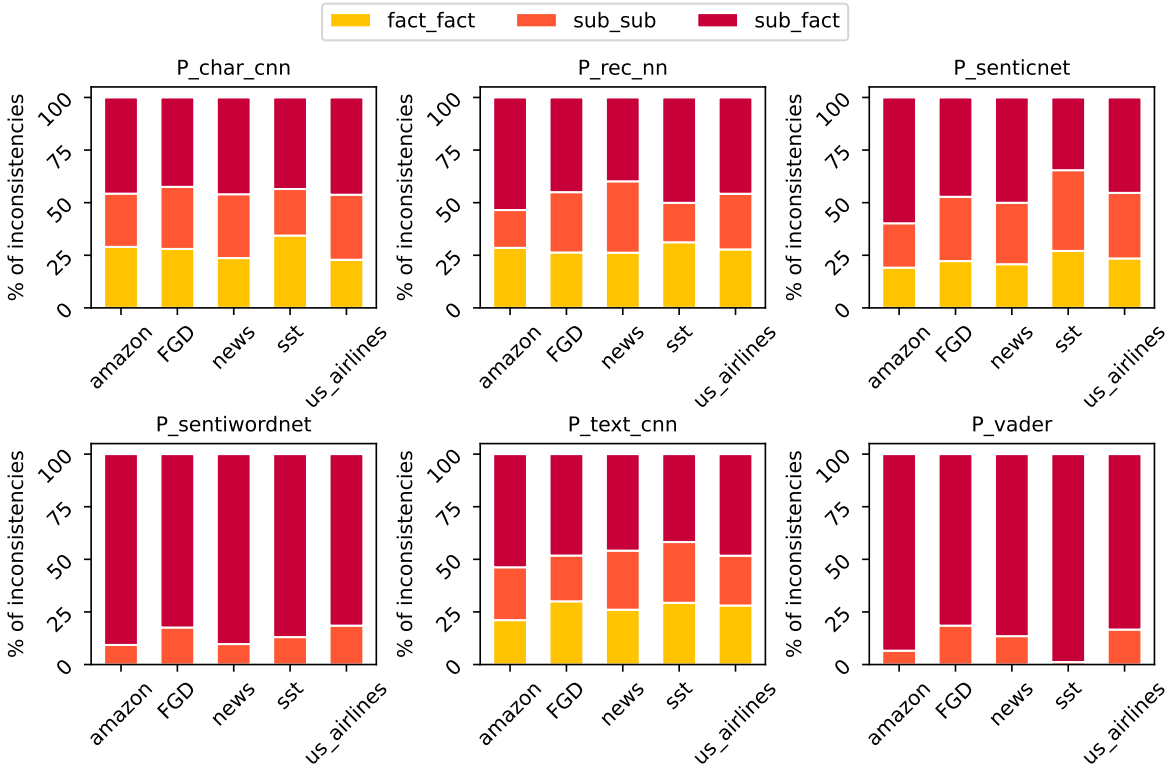


Figure 3.8: Inconsistencies and polar fact

For this, we first extract two sub-datasets from the original ones by filtering polar facts from opinionated documents using the word-lexicon-based method $P_{\text{sentiwordnet}}$. We first eliminate objective documents based on the ground truth, i.e., documents with Neutral polarity, then we consider the documents classified as Positive/Negative opinionated, and the misclassified Positive and Negative documents as Neutral polar facts. After that, we calculate the proportion of intra-tool inconsistencies for different tools on the opinionated/fact sub-datasets (Y-axis).

The results are displayed in Figure 3.8, where sub_sub represents the proportion of inconsistencies between opinionated documents, $fact\_fact$ represents the proportion of inconsistencies between polar facts, and sub_fact the proportion of inconsistencies between polar facts and opinionated documents.

We notice that many inconsistencies are between polar fact and opinionated documents in all tools/datasets. However, we have no inconsistencies between polar facts in lexicon-based methods because those tools classify polar facts as Neutral most of the time.

For other methods, we notice a difference in the proportion of inconsistencies. For instance, on P_{char_cnn}, $recn\_nn$, and $text\_cnn$, we have more inconsistencies between polar facts than between opinionated documents with proportions of 29.8%, 29.6%, and 32.2%, respectively. On $P_{senticnet}$, we have more inconsistencies between opinionated documents with a proportion of 27.3%.

Summary

This experiment shows that most inconsistencies occur between polar facts and opinionated documents, which motivates us to do more experiments and verify which of them is likewise closer to the ground truth.

3.3.4.2 Inconsistencies and Accuracy

In this section, we show the relationship between inconsistency and accuracy in different tools. For this, we calculate on each dataset the accuracy, the mean intra-tool, and inter-tool inconsistency rate following Equation 3.11. The results are summarized in Table 3.7.

We observe a high accuracy on the machine learning-based tools P_{char_cnn}, P_{text_cnn} and P_{rec_nn} (mean accuracy of 56.1%, 62.7696%, and 55.42% respectively) compared to the lexicon based methods where we notice a mean accuracy of (42.23% on $P_{sentiwordnet}$, 49.42% on P_{vader}, and 42.8% $P_{senticnet}$).

We notice that a low intra-tool inconsistency does not always imply a good accuracy since we have a low intra-tool inconsistency on lexicon based methods and a low accuracy.

3.3.5 Hyperparameters and Inconsistency

In this section, we focus on studying the impact of the learning hyper-parameters on the intra-tool inconsistency and the accuracy of the CNN.

For that, we consider the basic CNN described in [44], that we train each time by varying one of the hyper-parameters: the training dataset, the embedding type, the batch size, the cross validation folds, the learning rate, the dropout probability and the number of training epochs following the values presented in Table 3.8.

We answer the following questions:

(1) Is intra-tool inconsistency present for all CNN configurations?

(2) What is the best configuration that respects both accuracy and consistency?

(3) Which training dataset guarantees a general and accurate model?

3.3.5.1 Inconsistency and training dataset

In this experiment, we evaluate the robustness of the learned model based on the training dataset. We trained the model each time on the original, not extended, datasets (*Amazon*, *News* , *SST*, *FGD*, and *US airlines*) and tested the model on our benchmark. This experiment allows us to verify the generalization of the model on the extended dataset and the other datasets.

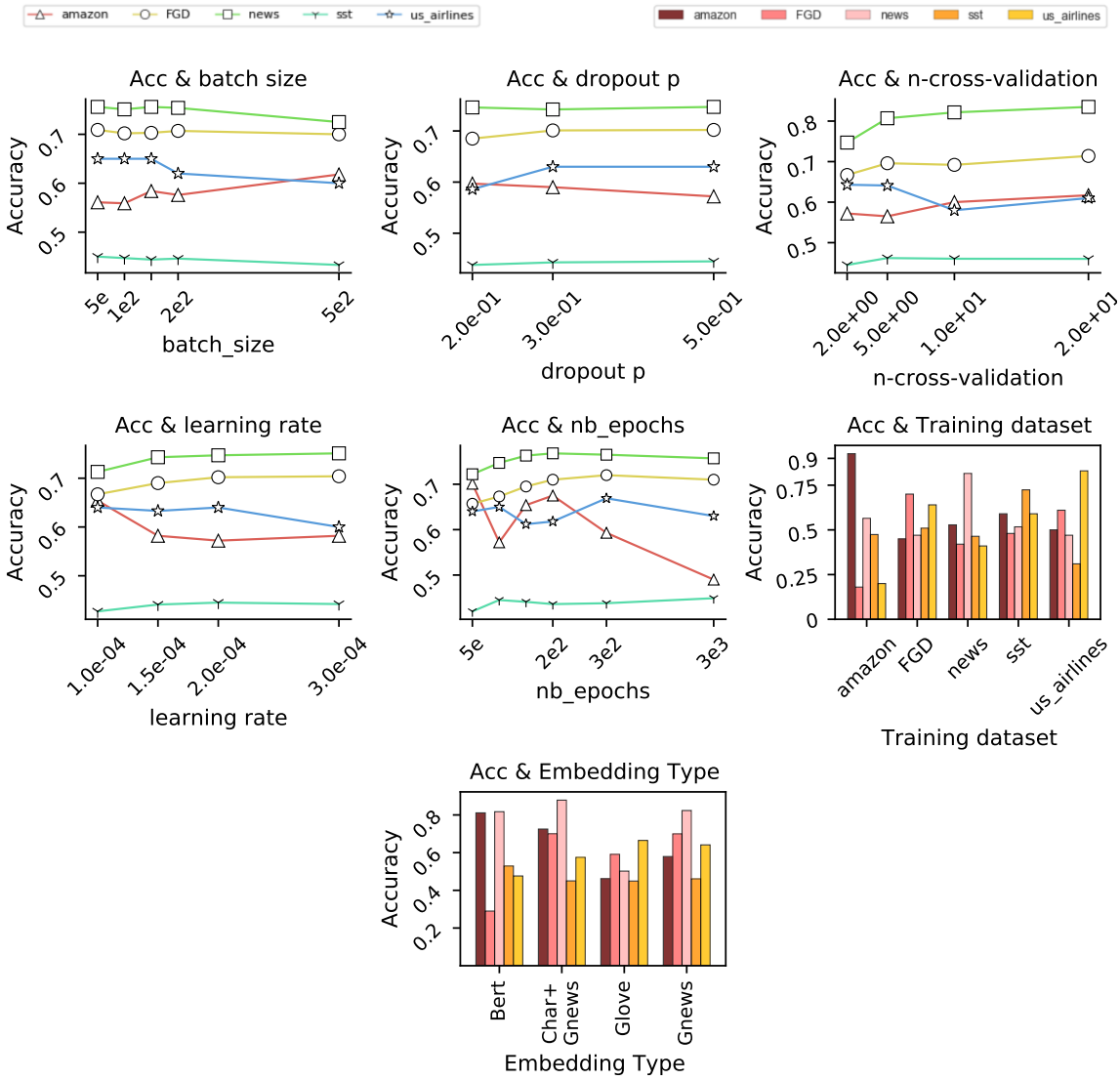


Figure 3.9: Accuracy and learning hyperparameters

Hyperparameter	Values					
Training dataset	Amazon	News	SST	FGD	US Airlines	-
Embedding type	word2vec (Google news vectors)	word2vec (Glove)	BERT	Char + word2vec	-	-
batch size	50	100	150	200	300	500
n-cv	2	5	10	-	-	-
dropout p	0	0.1	0.2	0.5	-	-
lr	10^{-4}	$1.5.10^{-4}$	2.10^{-4}	3.10^{-4}	-	-
nb epochs	50	100	150	200	300	3000

Table 3.8: Hyperparameters values

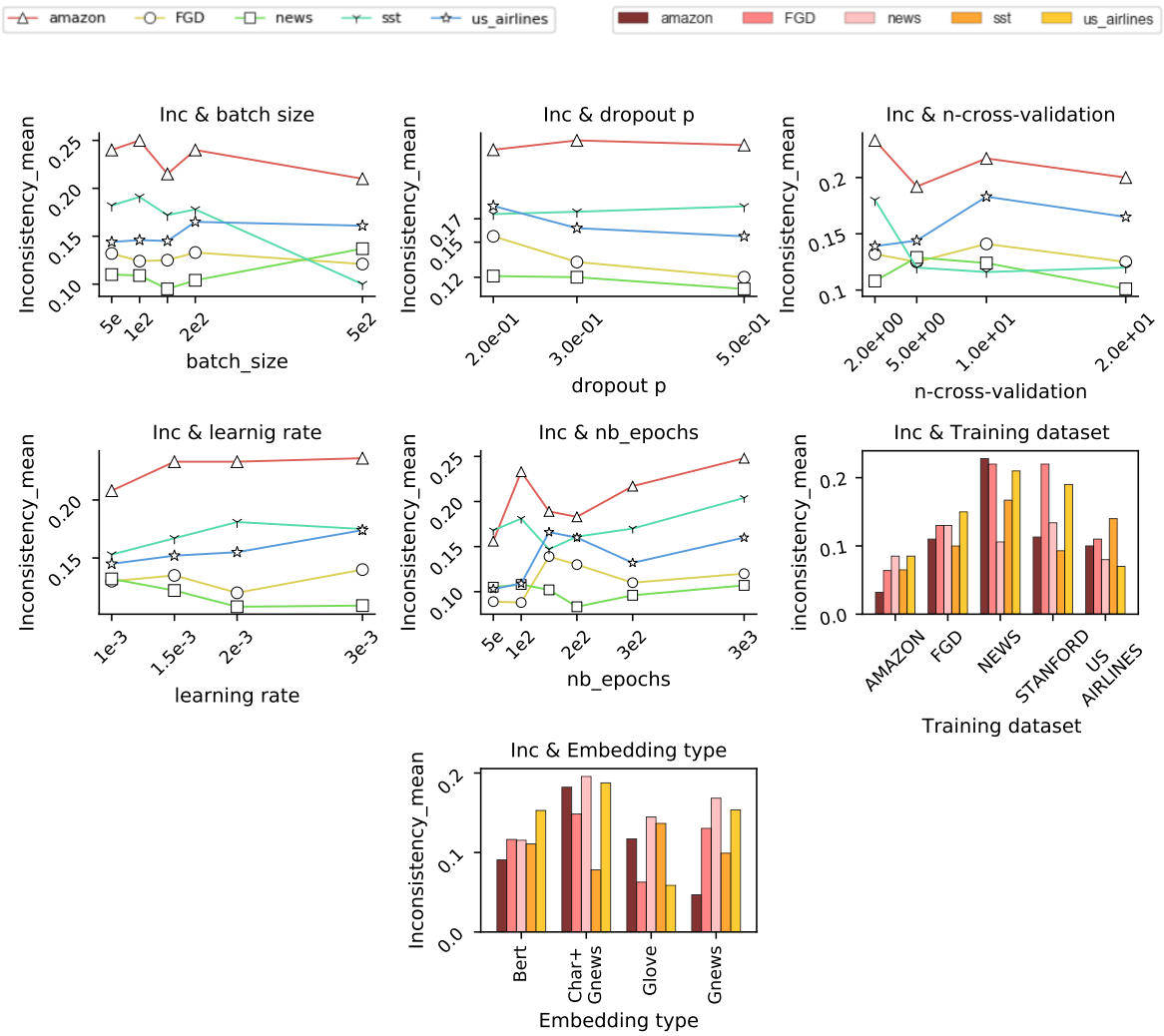


Figure 3.10: Inconsistency and learning hyperparameters

The results are presented in Figure 3.10 (Inc & training dataset) and Figure 3.9 (Acc & training dataset). We observe a mean accuracy of 50% on the models trained with the Amazon dataset (92.6% on *Amazon*, 56% on *News*, 47% on *SSt*, 18% on *FGD* and 20% *US airlines* datasets), 55.4% on the model trained with the *FGD* dataset, 52.8% on the models trained with the *News* dataset, 58.2% on the model trained with *SST* and 54.4% on the model trained with *US airlines*.

For the inconsistency rate, the *Amazon* dataset has a mean inconsistency of 0.066, while The *FGD*, *News*, *SST* and *US airlines* -trained model achieve 0.124, 0.186, 0.15, and 0.1, respectively.

Summary

We notice that the model trained with amazon is more general, since it leads to good accuracy and inconsistency when applied to other datasets (i.e., news and reviews), but not on the tweets datasets, which contain many abbreviations and errors, and need models specifically trained for them.

3.3.5.2 Inconsistency and embedding type

In this experiment, we check the robustness of the model with different embedding types: a pre-trained version of BERT as an embedding layer of the CNN, a word embedding using google news vectors, a word embedding using Glove and an embedding combining words and characters. The results of this experiment are presented in Figure 3.10 (Inc & embedding type) and 3.9(Acc & embedding type).

We notice a mean accuracy of 58.7% on the model with the BERT embedding, a mean accuracy of 66.6% on the model with word and character embeddings, a mean accuracy of 53.4% on Glove and a mean accuracy of 64.1%% on the model with google news vectors word embedding.

We notice a high accuracy on social media data (*FGD* and *US airlines* datasets) on models that use character embedding and word2vec embedding.

We notice also a mean inconsistency degree of 0.117 on BERT, 0.158 on models with character and word embeddings. On the models with a word embedding layer that uses Glove, we notice a mean inconsistency of 0.104.

Summary

We notice that the model using the pre-trained BERT as embedding is not adapted to Twitter data. Unsurprisingly, the use of the pre-trained BERT for embedding has improved the model's generalization on the *News* and *Amazon* datasets.

We can explain this generalization by the context-dependent nature of BERT that allows us to generate different embeddings of the word depending on the context.

The best inconsistency score (the lowest) was obtained on the model that uses the pre-trained Glove embeddings.

3.3.5.3 Inconsistency and batch size

In this experiment, we train the model described in [44] on the *News* dataset with a batch size of : 50, 100, 150, 200, 300, and 500. The results of this experiment are presented in Figure 3.10 (Inc & batch size) and 3.9(Acc & batch size).

On the *News* dataset, we observe that the accuracy decreases when the batch size increases, (from an accuracy of 0.756 for a batch size of size 50 to an accuracy of 0.724 for a batch size of 500).

On the *Amazon* dataset, the accuracy increases with the batch size (accuracy of 0.56 on the model with batch size of 50 to an accuracy of 0.68 for a batch size of 500).

On the *SST* dataset, the accuracy slightly drops from 0.455 to 0.433. We also observe that, contrary to the other datasets, inconsistency increases on the *News* dataset.

Summary

The optimal batch size is 300. A smaller batch size does not help the model generalize well as good inconsistency and accuracy results are only observed for the training dataset.

3.3.5.4 Inconsistency and cross validation.

In this experiment, we evaluate the impact of the n-cross validation on both the inconsistency and accuracy. We trained the model [44] on the *News* dataset and we varied n using the values: 2, 5, 10 and 20.

We observe that the accuracy increases with the cross validation size (n) on all datasets, while the inconsistency decreases with n on the *News* and *Amazon* datasets.

Summary

Cross validation increases the accuracy, consistency and generalization of the model. We recommend using 10-cross validation to optimize the consistency/accuracy trade-off.

3.3.5.5 Inconsistency and dropout probability

In this experiment, we verify the impact of the dropout probability on the accuracy, inconsistency and generalization of the model. To achieve this, we train our model and vary the dropout probability following the values: 0.2, 0.3 and 0.5. The results are presented in Figures 3.10 (Inc & dropout p) and 3.9 (Acc & dropout p).

We notice that the inconsistency on the datasets decreases when the dropout probability increases. The accuracy increases with the dropout probability on all datasets except the *Amazon* dataset.

Summary

The inconsistency decreases with the dropout probability.

3.3.5.6 Inconsistency and learning rate

To evaluate the impact of the learning rate on inconsistency and accuracy, we train our model on the *News* dataset, and we vary the learning rate values following the values: $1e^{-4}$, $1.5e^{-4}$, $2e^{-4}$, and $3e^{-4}$. The results are presented in Figures 3.10 (Inc & learning rate) and 3.9 (Acc & learning rate).

We observe that when the learning rate increases, the inconsistency decreases on the *News* dataset and increases on the others datasets.

We also observe that the accuracy increases with learning rate on the datasets *News*, *FGD*, and *SST* and decreases on the datasets *Amazon* and *US airlines* datasets.

Summary

The learning rate affects both accuracy and inconsistency since models with a low learning rate are more general and consistent.

3.3.5.7 The number of epochs

In this experiment, we verify the influence of the number of epochs on the inconsistency and the accuracy. We train the model on numbers of epochs following the values: 50, 100, 200, 300 and 3000, then test the model using our benchmark. The results in Figure 3.9 (Acc & nb_epochs) show that the accuracy of the model on the datasets increases then decreases as the number of epochs increases, which means that the model lost its generalization when trained with a high number of epochs (overfitting).

As for the inconsistency, Figure 3.10 (Inc & nb_epochs) shows that it first decreases then increases which confirms the overfitting explanation.

Summary

Even if training the model on a high number of epochs may improve the training accuracy of the model, this makes it lose in terms of intra-tool consistency and generalization.

3.3.5.8 Discussion

From the findings of the previous experiments, we observe that the inconsistencies are present on different CNN configurations.

We also notice that inconsistencies depend on the subjectivity of the document and most inconsistencies occur between polar facts and opinionated documents.

To verify which of the two types is closer to the ground truth, we calculate the accuracy in the sub-datasets of opinionated documents and facts.

	P_{text_cnn}	P_{char_cnn}	$P_{senticnet}$	$P_{sentiwordnet}$	P_{rec_nn}	P_{vader}	MV
Amazon	**84.4%**	**49.26%**	**60.7%**	**42.8%**	**54.5%**	**57.3%**	66.7%
News	**50.6%**	**51.4%**	**53.67%**	**33.48%**	**47.4%**	**47.2%**	**57.4%**
SST	45.9%	**77.3%**	**47.9%**	**49%**	**66%**	**51.5%**	63.4%
FGD	49.2%	**57.5%**	**30.9%**	**42.07%**	**71.05%**	**44.1%**	58.7%
US airlines	64.6%	**68.1%**	24.5%	**46.11**	**65.4%**	**50.3%**	65.5%

Table 3.9: Accuracy after resolving inconsistencies using MV

We found that polar facts are more accurate on P_{char_cnn} with an accuracy of 62% for polar facts and 57.2% for opinionated documents, while in other tools we notice a higher accuracy on opinionated data (74.5%, 68.4%, 73.5% for opinionated data and 55.7%, 56.1% and 64.4% for polar facts on P_{rec_nn}, $P_{senticnet}$ and P_{text_cnn}, respectively).

In this case, we can use the document nature as a feature when resolving inconsistency.

These findings motivate us to study the impact of resolving inconsistency on accuracy. We display in Table 3.9 the accuracy of tools after resolving intra-tool inconsistency on different tools using majority voting (MV) and both intra-tool and inter-tool inconsistency (the Column MV).

We observe an important improvement of the accuracy when resolving intra-tool inconsistency. This may be explained by the fact that the tools prediction may be erroneous on some documents and correct on others.

Hence, resolving intra-tool inconsistency by unifying the polarity in the analogical set may increase the accuracy in most cases.

We observe an accuracy degradation when resolving inconsistencies on the *SST*, *FGD*, and *Amazon* datasets, because the tools are not compatible in terms of accuracy on these datasets. To overcome this problem and guarantee the accuracy improvement, we recommend to apply majority voting between tools compatible in terms of accuracy.

Few works have exploited the inconsistency to improve the accuracy of classification, such as the work in [2], where the authors minimize inter-tool inconsistency of various labeling functions based on the accuracy, correlation and label absence.

These solutions outperform majority voting only for a low setting of redundancy (i.e., when the number of tools is low), which confirms that the inconsistency problem is not yet resolved.

However, these works do not consider both intra-tool and inter-tool inconsistencies, which would lead to better results.

Since the inconsistency problem has been wildly studied in data management [121]–[123] and the fact inference field [88], [124], [125], where has been made tangible progress, we suggest referring to methods from fact inference by considering workers as tools in the formalization to resolve inconsistency and improve accuracy [124], [125].

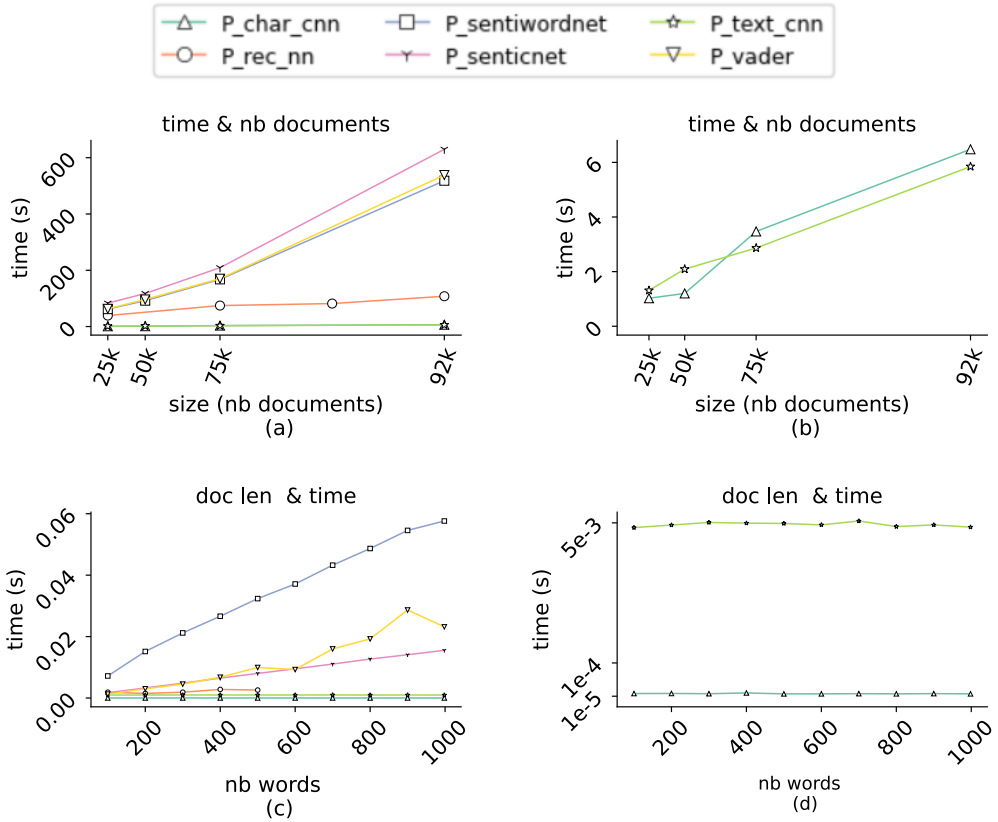


Figure 3.11: Scalability of tools in time (number of documents and document's size)

3.3.5.9 Scalability experiments

In this experiment, we evaluate the scalability of tools in dataset and document size.

Figures 3.11(a-b) show execution time when we vary the data size: 25%, 50%, 75%, and 100% of the entire benchmark. (Figure 3.11(b) focuses on the performance of P_{char_cnn} and P_{text_cnn} only.)

We observe, that for lexicon-based methods and P_{rec-nn}, the time scales (almost) linearly with the size. Machine learning methods P_{text_cnn} and P_{char_cnn} are faster than the other tools. This is explained by the fact that these tools perform prediction in parallel (prediction by batch), which accelerates the prediction process.

We also evaluate the scalability of tools in response time when varying the document size (number of words), using a synthetic dataset sampled from the word corpus of nltk[6].

The results are presented in Figures 3.11(c-d), where the x-axis is document size expressed as number of words and y-axis is execution time. (Figure 3.11(b) focuses on the performance of P_{char_cnn} and P_{text_cnn} only.)

We observe that execution time scales linearly with the document size for all meth-

[6]https://pypi.org/project/nltk/

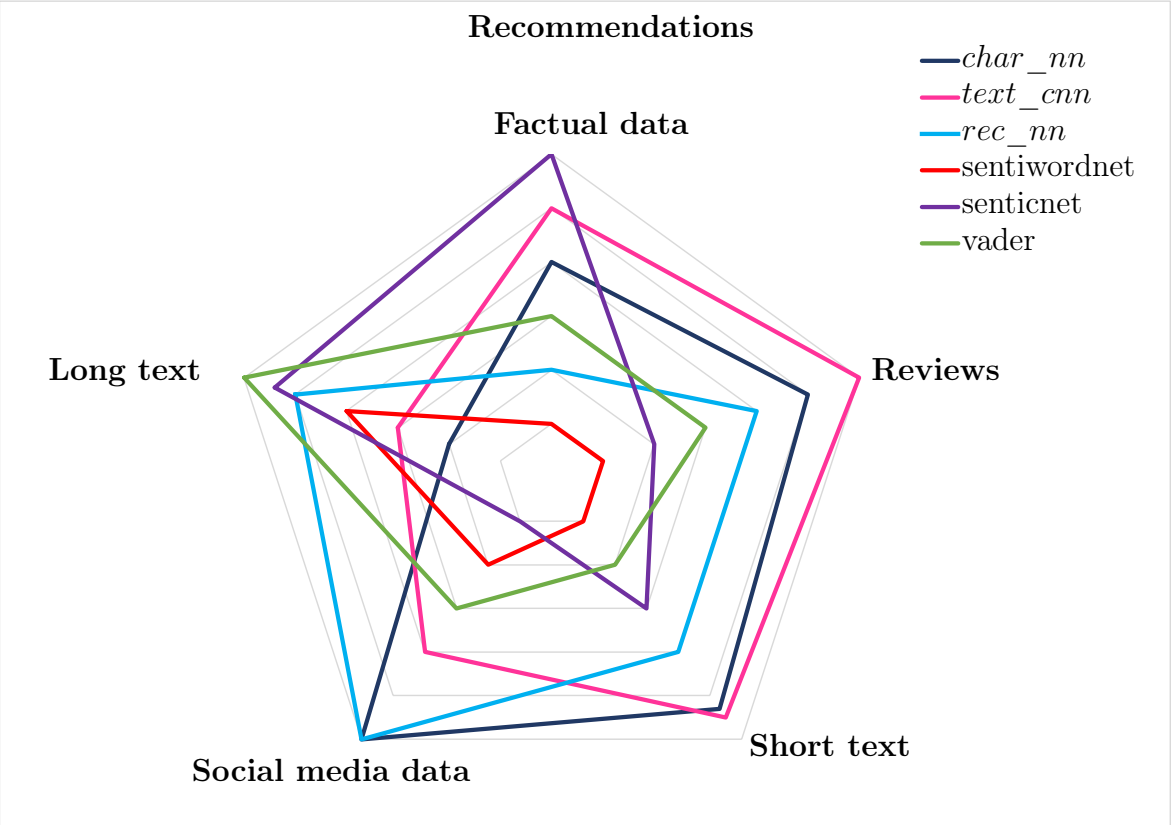


Figure 3.12: Recommendations to choose the sentiment analysis tool following text type (long text, short text, social media data, reviews, and factual data). It measures the performance of the tool given a data type.

ods except for the machine learning ones: these methods use a fixed-size embedding, and they then crop or pad the text (if it is too long or short, respectively), which makes them inaccurate for long text.

Summary

In light of this experiment, we suggest parallelizing the prediction process to improve scalability and time performance.

In the next section, we present a set of recommendations to choose the most relevant sentiment analysis for a given scenario.

3.4 Recommendations

The findings of the previous experiments have demonstrated that the inconsistencies depend on different factors such as text subjectivity, text structure, tools type, and tools configuration. Moreover, as we can observe in Table 3.7, tools performance, i.e., inconsistency and accuracy, changes following the data type.

Figure 3.12 presents the tools' performances depending on data type: short text,

long text, social media data (tweets), reviews, and factual data.

In light of these observations, we abstract a guideline for selecting the relevant sentiment analysis for a given scenario:

- For short text, we recommend CNN based methods. Short texts have few sentiment features, which can not be effectively handled by word-lexicon or concept based methods.
- From the explainability point of view, we recommend lexicon based methods, especially those based on concept lexicons, thanks to their performance in terms of accuracy.
- For streaming data with long text and a large window, we recommend an ensemble of lexicon-based methods compatible in terms of accuracy and resolve inter-tool inconsistency between them to improve accuracy.
- For small window sizes (i.e., number of documents < 1000 per δt), we recommend CNN.
- For social media data, both character and word embedding are recommended, since we can train the model to be fault tolerant. For reviews, word embedding is enough and for news and factual data, we recommend BERT embedding.
- To resolve inconsistencies and improve accuracy we suggest using truth inference methods
- Inconsistencies are influenced by the analogical set structure and the document nature (fact or opinionated).
- Our insights have implications for various use cases, such as crowdsourced fact checking, where we can accurately classify workers, remove inconsistent labels, or alleviate biased labeling.
- The scalability experiments point to exciting research opportunities for the data management community to improve the scalability properties of existing solutions to serve the machine learning and NLP communities better.

3.5 Conclusion

This chapter presents the first large study on data quality for sentiment analysis tools, reveals the causes and factors that influence the inconsistencies, describes the benchmark we built, and provides a set of tips and recommendations to select the most appropriate tool for a given scenario.

Our study covers statistical, structural, and semantic analysis for the inconsistency problem. It shows that the inconsistency problem is not yet resolved and argues for the potential improvement obtained on accuracy when resolving intra-tool and inter-tool

inconsistencies. Moreover, our analysis offers several findings of interest to the machine learning, NLP, and data management communities.

To the best of our knowledge, no previous research work has explored the problem of resolving both intra-tool and inter-tool inconsistencies on tools accuracy. Thus, in the next chapter, we investigate the impact of resolving inconsistency on accuracy.

Chapter 4

Study on Inconsistency Resolution in Sentiment Analysis Tools

The findings in chapter 2 and the state-of-the-art works presented in chapter 1 lead us to conclude that inconsistencies are frequent in sentiment analysis tools and need to be resolved since they are harmful and warn that tools are making prediction errors.

This chapter presents first SAQ (Sentiment analysis quality), a framework for inconsistency resolution on sentiment analysis tools that resolves intra-tool and inter-tool inconsistencies to enhance sentiment extraction quality, i.e., accuracy and consistency. SAQ uses Markov logic network to model the documents, tools, and relations between them. The intuition of SAQ is to seek convergence to the golden truth by resolving intra-tool and inter-tool inconsistencies based on the two consensuses: a document has a unique polarity, and the semantically equivalent documents have the same polarities. Consequently, resolving inconsistencies minimizes the number of incorrect labels.

After that, we conduct the first study that evaluates the effect of resolving both intra-tool and inter-tool inconsistency on sentiment analysis tools' accuracy. To the best of our knowledge, we are the first to study the consistency of sentiment analysis tools on the sentence level. We compare SAQ to three state-of-the-art methods for resolving inconsistency using the benchmark presented in chapter 3, study the effect of resolving inconsistency on accuracy and determine a guideline to choose tools to guarantee accuracy improvement and avoid the propagation of erroneous labels.

In the remainder of the chapter, we present in Section 1 SAQ modeling to resolve both intra-tool and inter-tool inconsistency. Then, in Section 2, we present our empirical study on inconsistency resolution and discuss the results.

.

4.1 SAQ: Sentiment Analysis Quality

Many research works have been interested in inconsistency resolution, including ensemble methods (Section 2.3.1), weak supervision methods (Section 2.3.3), and truth inference in crowdsourcing (Section 2.3.2).
However, all these works consider only one type of inconsistency (intra-tool [3] or inter-tool [2]), and none of them have considered resolving the two types of inconsistencies: intra-tool and inter tool.

Hence, in this section, we present SAQ, a.k.a, Sentiment Analysis Quality, the first method to resolve both types of inconsistencies, i.e., intra-tool and inter-tool on sentiment analysis tools.
In the beginning, we present an overview of SAQ, then we describe each component, and finally, we explain the reasoning process of SAQ on a running example (Example 1.1).

4.1.1 Overview

SAQ is a framework that allows resolving intra-tool and inter-tool inconsistencies between tools and documents based on the consensus that semantically equivalent documents have the same polarity and each document has a unique polarity. It improves polarity prediction accuracy by reducing the number of incorrect labels and converging to the ground truth.

Modeling polarity inconsistencies is complex because it involves tools, documents, and relationships between them represented by semantic similarity and inconsistencies. Furthermore, the presence of inconsistencies requires using a modeling framework that handles uncertainty over relational models. Hence the choice of MLN to model SAQ. This choice is justified by the features of MLN that allow defining probabilistic distributions over relational structures and handling uncertainty in modeling.

Figure 4.1 shows an overview of SAQ that takes as input a dataset clustered into analogical sets and sentiment analysis tools and provides an output set of polarity labeled documents with uncertainty scores. SAQ contains the following modules:

- **Semantic module.** It contains the essential knowledge for reasoning and inconsistency resolution.
- **Weights calculation module.** It allows to rank tools based on their inconsistency (intra-tool and inter-tool).
- **Polarity inference module.** It uses the MLN inference procedure to infer the polarity that minimizes the inconsistency.

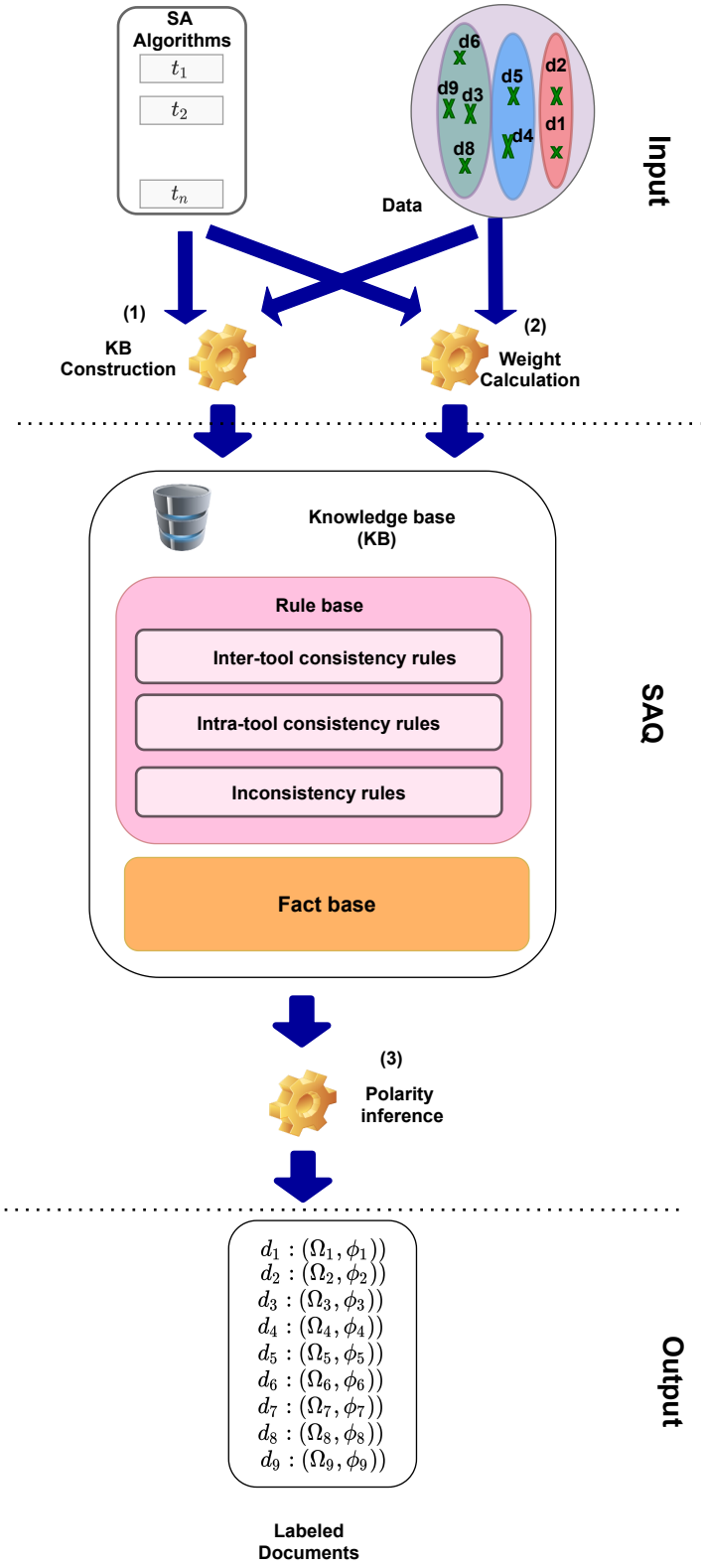


Figure 4.1: SAQ overview

In the following, we describe each module in detail and explain the reasoning process

in SAQ.

4.1.2 Semantic Module

The semantic module contains the necessary vocabulary to model tools, documents, and relations, rules to model and detect inconsistencies and a set of facts. It is a knowledge-base $KB =< R, F >$, where:

1. R is a *set of rules* (FOL formulas) defining the vocabulary of our application which consists of concepts (sets of individuals) and relations between them.
2. F is a *set of facts* representing instances of the concepts or individuals defined in R.

4.1.2.1 Vocabulary

We represent each document by the concept *Document*, each polarity function a.k.a, sentiment analysis tool, by its symbol and the polarity that it attributes to the *Document*. For instance, $P^{+}_{rec\_nn}(d_i)$, $P^{0}_{rec\_nn}(d_i)$, and $P^{-}_{rec\_nn}(d_i)$ represent respectively the polarities (+, 0, -) attributed to the $Document(d_i)$ by the polarity function P_{rec_nn}.

Each *Document* is $Positive$, $Negative$, or $Neutral$. The document's polarities are represented respectively by the concepts $IsPositive$, $IsNegative$, and $IsNeutral$.

We also have the relation *sameAs* that represents the semantic similarity between documents in the input dataset. For instance, $sameAs(d_i, d_j)$ indicates that the documents $Document(d_i)$ and $Document(d_j)$ are semantically equivalent.
Table 4.1 resumes the vocabulary of SAQ to model Example 1.1.

Predicates	Semantics	Truth values
$Document(d_i)$	The document to be annotated	True for each document to be annotated
$P^{+}_{rec\_nn}(d_i)$, $P^{-}_{rec\_nn}(d_i)$, $P^{0}_{rec\_nn}(d_i)$ $P^{+}_{Senticnet}(d_i)$, $P^{-}_{Senticnet}(d_i)$, $P^{0}_{Senticnet}(d_i)$ $P^{+}_{Sentiwordnet}(d_i)$, $P^{-}_{Sentiwordnet}(d_i)$, $P^{0}_{Sentiwordnet}(d_i)$ $P^{+}_{Vader}(d_i)$, $P^{-}_{Vader}(d_i)$, $P^{0}_{Vader}(d_i)$	The polarity function of the tool that models the polarity Ω assigned by the tool t_k to the document d_i with $\Omega \in \{+, -, 0\}$.	True if the tool t_k assigns the polarity Ω to the document d_i
$isPositive(d_1)$ $isNegtive(d_i)$ $isNeutral(d_i)$	The polarity of the document.	Inferred
$sameAs(d_i, d_j)$	The semantic similarity between documents d_i and d_j	True if d_i and d_j are semantically equivalent

Table 4.1: Vocabulary of SAQ applied to Example 1.1

4.1.2.2 Rules

We define two types of rules in SAQ over the vocabulary, *Inference rules* and *Inconsistency rules*.

The *inference rules IR* allow deriving the implicit instances. They model the quality of the polarity at intra-tool and inter-tool levels. They are soft rules that add an uncertainty layer to different polarity functions based on the tool inconsistency. We used two types of inference rules: intra-tool and inter-tool consistency rules.

The *intra-tool consistency rules* assert that all the semantically equivalent documents should have the same polarity. They are defined as[1]:

$$IR1 : IsPositive(d_i) \leftarrow sameAs(d_i, d_j) \wedge IsPositive(d_j)$$
$$IR2 : IsPositive(d_j) \leftarrow sameAs(d_i, d_j) \wedge IsPositive(d_i)$$

The rules $IR1$ and $IR2$ are examples of intra-tool inconsistency rules and denote that if two documents d_i and d_j are semantically equivalent (expressed with *sameAs* relation), they got the same polarity, which translates the intra-tool consistency defined in equation 3.1.

The *sameAs* relation is symmetric, reflexive, and transitive. We express the symmetry by duplicating the rule for both documents of the relation (rules $IR1$ and $IR2$ instead of only one rule).

For instance, in Example 1.1, when applying the rule $IsNeutral(d_i) \leftarrow sameAs(d_i, d_j) \wedge IsNeutral(d_i)$ on the relation $sameAs(d_1, d_2)$ and the instances $IsNeutral(d_1)$ and $IsNegative(d_2)$, we entail the new instance $IsNeutral(d_2)$ while when applying the rule $IsNegative(d_i) \leftarrow sameAs(d_i, d_j) \wedge IsNegative(d_j)$ the instance $IsNegative(d_1)$ is entailed.

The transitivity is handled in the instantiation step (algorithm 7). We ignore the reflexivity because it does not infer additional knowledge and provides cycles in reasoning.

The *inter-tool consistency rules* model the inter-tool consistency described in equation 3.3 by assuming that each function gives the correct polarity to the document.

For each tool, we associate the following rules by replacing P_{t_k*} with the polarity function of the tool:

$$IR3 : IsPositive(d_i) \leftarrow P^{+}_{t_k}(d_i)$$
$$IR4 : IsNegative(d_i) \leftarrow P^{-}_{t_k}(d_i)$$
$$IR5 : IsNeutral(d_i) \leftarrow P^{0}_{t_k}(d_i)$$

These rules are soft. They allow us to handle inconsistencies and use weights to rank rules. The idea behind this modeling is that if the inter-tool consistency is respected, all tools will attribute the same polarity to this document; otherwise, the document will have different polarities (contradicted polarities). To represent this contradiction,

[1] for the sake of clarity, we omitted the predicate $Document(d_i)$ in all logical rules

we define the inconsistency rules:

The *inconsistency rules ICR* are hard rules representing the disjunction between polarities and assert that each document has a unique polarity. Inconsistency rules are defined as:

$$ICR1 : \neg IsNegative(d_i) \wedge \neg IsNeutral(d_i) \leftarrow IsPositive(d_i)$$

$$ICR2 : \neg IsPositive(d_i) \wedge \neg IsNeutral(d_i) \leftarrow IsNegative(d_i)$$

$$ICR3 : \neg IsPositive(d_i) \wedge \neg IsNegative(d_i) \leftarrow IsNeutral(d_i)$$

These rules are hard and generate negative instances that discover inconsistencies used in inference.

For instance, consider the following instances from Example 1.1 $P_{Senticnet^+}(d_4)$ and $P_{Sentiwordnet^-}(d_4)$ from the motivating example. By applying the inter-tool consistency inference rules, we entail: $IsNegative(d_4)$ and $IsPositive(d_4)$. However, F appears consistent even it contains polarity inconsistencies. When applying the inconsistency rules by entailing : $\neg IsPositive(d_4)$, $\neg IsNeutral(d_4)$, $\neg IsNegative(d_4)$, the inconsistencies become explicit. In the following, we describe the process we followed to build the knowledge base.

4.1.2.3 Knowledge base construction

Our data are first saved in a relational database, where each table represents a concept, and the table content represents the concept's instances. The knowledge base construction process is described in Algorithm 7.

This algorithm aims to add the prior knowledge needed in the reasoning to the set F and add the rules to the set R . The prior knowledge is: the documents, polarities attributed to documents, and the semantic similarity between documents represented by the *SameAs* predicate.

4.1.3 Weights Calculation Module

The learning procedure of MLN is supervised and requires correctly labeled data (polarity ground truth). However, since we do not have quality labeled data, we propose to weight the rules based on tools inconsistency rate by attributing higher weights to less inconsistent tools.

To calculate the weights, we consider the intra-tool inconsistency rate in equation 3.11 and the inter-tool inconsistency in equation 3.12.

Algorithm 7 KB Construction

Input: D: Dataset, Π : Set of tools
Output: F:Set of generated Facts
R: Set of rules

procedure Instantiating

//**Step1:** Add all Polarities attributed to documents
for each $P_{t_k} \in \Pi$:
for each $d_i \in D$: $F.add(P_{t_k^+}(d_i))$
$F.add(P_{t_k^-}(d_i))$
$F.add(P_{t_k^0}(d_i))$

//**Step2:** Add sameAs relations
clusters = groupByClusterId(D)
for each *cluster* $\in$ *clusters* :
for $i \in$ *{0,. . . , len(cluster)}* :
for $j \in$ *{i+1 ,. . . , len(cluster)}* :
if $SameAs(d_i, d_j) \notin F$: $F.add(SameAs(d_i, d_j))$

//**Step3:** Add rules
for each $P_{t_k} \in \Pi$:
R.add($isPositive(d_i) \leftarrow P^+_{t_k}(d_i)$)
R.add($isNegative(d_i) \leftarrow P^-_{t_k}(d_i)$)
R.add($isNeutral(d_i) \leftarrow P^0_{t_k}(d_i)$)

//**Step4:** Add intra-tool consistency rules

//**Step5:** Add inconsistency rules

return $< F, R >$

Since we represent the consistency of one tool by three rules, with a rule for each polarity, we define the intra-tool inconsistency of a tool P_{t_k} given a polarity Ω over a dataset D by:

$$\begin{aligned} &\forall P_{t_k} \in \Gamma\,, inc^{\Omega}_{intra}(D, P_{t_k}) = \sum_{l=1}^{m} \frac{card(S^{\Omega}_l)}{m\, n_l(n_l - 1)} \\ &S^{\Omega}_l = \{d_i, d_j \in A_l | P_{t_k}(d_j) = \Omega \wedge P_{tk}(d_i) \neq \Omega\} \end{aligned} \tag{4.1}$$

with $\Omega \in \{+, 0, -\}$, n_l the number of documents in the analogical set A_l (see Definition 17), and m the number of analogical sets in the dataset D.

The inter-tool inconsistency of a tool P_{t_k} and polarity Ω over a dataset D by:

$$\begin{aligned} &\forall P_{t_k} \in \Gamma,\, inc^{\Omega}_{inter}(D, P_{t_k}) = \sum_{j=1}^{q} \frac{card(S'^{\Omega}_j)}{q(z-1)} \\ &s.t\, S'^{\Omega}_j = \{P_{t_{k'}} \in \Gamma | P_{t_k}(d_j) = \Omega \wedge P_{t'_k}(d_j) \neq \Omega\} \end{aligned} \tag{4.2}$$

with q the size of the dataset D and z is the number of polarity functions in Γ.

To calculate tools weights based on inconsistency, we aggregate intra-tool and inter-tool inconsistency scores given respectively by equation 4.1 and 4.2 where the highest weight is attributed to tools with the lowest inter-tool and intra-tool inconsistency.

We define $w^{\Omega}_{t_k}$, the weight of the representative rule of polarity function P_{t_k} given the polarity Ω as:

$$w^{\Omega}_{t_k} = \frac{\sum_{A \subseteq D} \lambda\, exp(-\phi\, inc^{\Omega}_{intra}(A, P_{t_k}))}{inc^{\Omega}_{inter}(D, P_{t_k})} \tag{4.3}$$

with ϕ and λ are parameters that we determine empirically. These weights are used to rank inter-tool consistency inference rules by assigning higher importance to the most consistent tools in the inference process.

4.1.4 Polarity Inference Module

This module allows us to infer the most appropriate polarity by applying the inference procedure of MLN [104] on KB. Inference in MLN includes two steps:

1. The grounding step samples all possible worlds based on the prior knowledge in KB and constructs a large weighted SAT formula for satisfiability calculation.

2. The search step seeks the best weight assignment to the SAT formula by maximizing the cost and favoring rules with higher weights, i.e., rules associated with the most consistent tools.

The Inference Module used the marginal inference algorithm that estimates the atoms' probability and returns the query answer with a probability score representing the polarity's uncertainty.

Next, we present the end-to-end reasoning process to resolve the inconsistency in SAQ.

4.1.5 MLN Based Inconsistency Resolution Process

To enhance the quality of sentiment analysis tools and resolve both intra-tool and inter-tool inconsistencies for accuracy improvement, SAQ follows the process described in Figure 4.1 and Algorithm 7.

SAQ takes into input a set of Documents (D) clustered into groups of semantically equivalent documents,i.e., analogical sets (see definition 17) and a set of polarity functions; then, it extracts the polarities from documents using the input polarity functions.

After that, SAQ builds the knowledge base KB by creating first the fact set F by adding the instances of Documents, the SameAs relations between every two documents of the analogical sets, and the polarity extracted by every polarity function from the documents.

It also constructs the set R by generating inter-tool inconsistency rules and adding the rest of the rules.

SAQ calculates the weights by computing the inter-tool and the intra-tool inconsistency rate following Equation 4.2 and Equation 4.1, then aggregating them following Equation 4.3.

Finally, we use the computed weights with the rules to infer the polarity that minimizes inconsistencies between soft rules. We note that the reasoning in SAQ favors the polarities given by the most consistent tools. The output of SAQ is a set of labeled documents with uncertainty scores.

4.2 Experiments

In this section, we study the effect of resolving inconsistency on the accuracy and determine the primary conditions on tools to guarantee an accuracy improvement. We evaluate SAQ empirically and compare it to state-of-the-art inconsistency resolution methods. It allows to answer the following questions:

- Does inconsistency resolution always lead to an accuracy improvement?

- Does resolving both intra-tool and inter-tool inconsistency better than resolving only inter-tool inconsistency?
- What are the conditions on tools that guarantee an accuracy improvement?

To answer these questions, we present first our experimental setup. Then we evaluate SAQ and compare it to the state-of-the-art methods for inconsistency resolution. Furthermore, we study the effect of resolving inconsistency on the accuracy in different settings, discuss the results, and quote the learned lessons.

For MLN inference, we used the Tuffy [4] implementation. All our experiments are implemented in python and java. We fix $\lambda = 3$ and $\phi = 4$

4.2.1 Experimental Setup

4.2.1.1 Datasets

Statistics	# Elements	# Positive	# Neutral	# Negative
$News\_heads$	1583	891	37	655
SST	3505	1481	640	1384

Table 4.2: Statistics on datasets.

We use the two augmented datasets, news headlines and SST, from the benchmark presented in Section 3.2. Table 5.2 summarizes the statistics of the used datasets.

4.2.1.2 Sentiment analysis tools

To study the effect of inconsistency resolution on sentiment analysis tools by varying the accuracy and the number of tools, we use sets of simulated polarity functions with different accuracy values.

To simulate the polarity functions, we consider a dataset annotated with polarity labels, then we sample a proportion of $1 - acc$ documents randomly and set their polarities randomly.

4.2.1.3 Truth inference method

To evaluate SAQ and study the effect of resolving inconsistency on the accuracy, we reduce the inconsistency resolution problem in sentiment analysis to a truth inference problem by considering tools as workers and the aggregated polarity as the inferred truth.

In this study, we use three methods from the state-of-the-art. The work in [88] has classified truth inference, a.k.a, inconsistency resolution methods, into three classes: probabilistic models, direct computing, and optimization (see Section 2.3.2). Hence, we use a representative method from each category:

Majority voting (MV) represents the direct and trivial computation to resolve inconsistency and infers truth. We perform majority voting to resolve the inconsistency between tools by considering as accurate polarity label, the most redundant polarity.

$$p_*(d_i) = \underset{\Omega \in \{+,0,-\}}{argmax} \sum_{p_{t_k} \in F} \mathbb{1}_{\{p_{t_k}(d_i)=\Omega\}}$$

Zencrowd (ZC) [93] is the primary method for inconsistency resolution using the probabilistic graphical model. It infers the most appropriate polarity given the probability of the tool. Formally

$$Pr(p_{t_k}(d_i)|q_{t_k}, p_*(d_i)) = q_{t_k}^{\mathbb{1}_{p_{t_k}(d_i)=p_*(d_i)}}.(1-q_{t_k})^{\mathbb{1}_{p_{t_k}(d_i)\neq p_*(d_i)}}$$

PM [126] represents the primary method that uses optimization to learn the quality of tools and infer the truth by optimizing the following objective

$$\min q^{t_k}, p_*(d_i) \sum_{t_k \in F} q^{t_k} \sum d(p_*(d_i), p_{t_k}(d_i))$$

with $p_*(d_i)$ is the inferred polarity of the document (truth), q^{t_k} the quality of the tool t_k, $\Omega \in +, 0, -$ represents the polarity, and $d(.,.)$ is the distance between two polarities.

4.2.2 Overall Performances

This experiment compares the obtained accuracy after inconsistency resolution between SAQ and the truth inference methods, a.k.a, inconsistency resolution methods ZC, PM, and MV. For this, we consider four polarity functions P_{t1}, P_{t2}, P_{t3}, and P_{t4}, that all have an accuracy of 60% on SST. Then we resolve the inconsistencies between them using the methods MV, SAQ, PM, and ZC. The experimental results are presented in Figure 4.2.

We observe that SAQ outperforms all inconsistency resolution methods with a significant accuracy improvement of 0.2 compared to 0.1 with the MV, 0.2 with PM and 0.05 with ZC.

These results indicate that resolving the two types of inconsistency, intra-tool and inter-tool inconsistencies, is beneficial and improves accuracy better than resolving only one type of inconsistency.

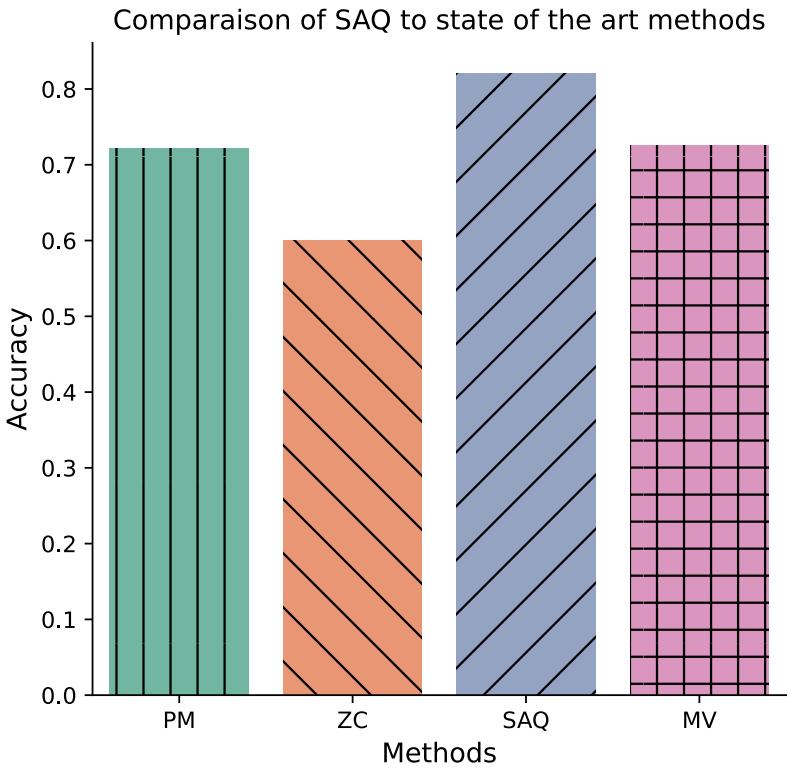


Figure 4.2: Comparison of SAQ to state-of-the-art methods for inconsistency resolution (PM, ZC, and MV)

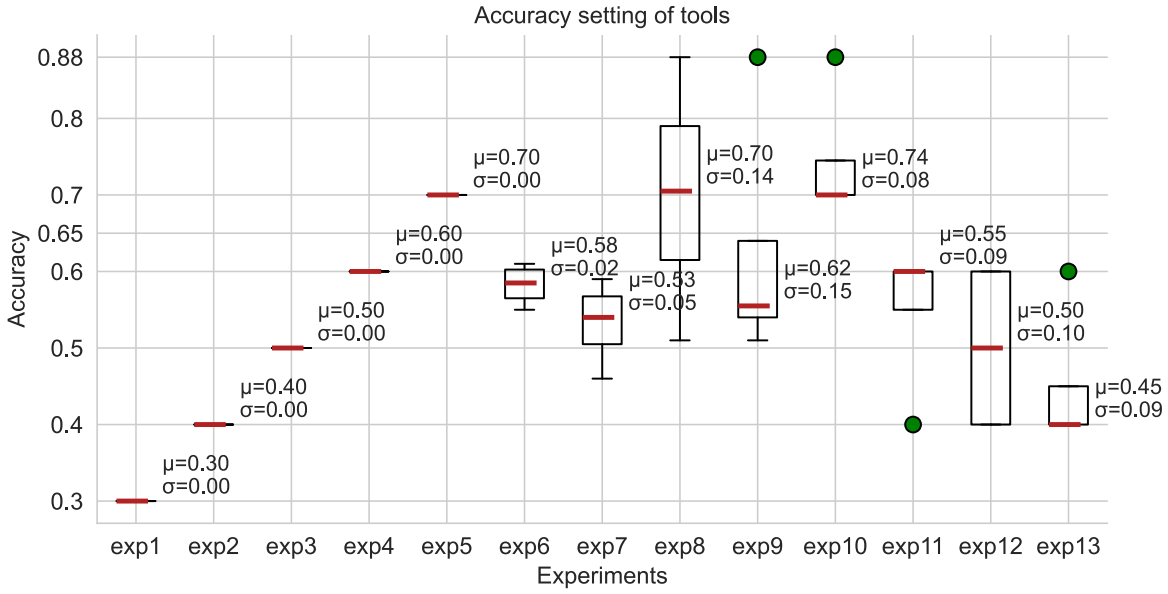


Figure 4.3: Statistical summary on used tools sets (x-axis: exp1 to exp13 are the sets of used tools, Y-axis: tools' accuracy)

4.2.3 The Primary Accuracy Effect of Tools

This experiment evaluates the effect of the primary accuracy of tools on the inconsistency resolution. Therefore, we run SAQ and MV on 13 sets ($exp1$ to $exp13$) of simulated polarity functions with different accuracy values. We then evaluate the effect of the primary accuracy of tools on the accuracy improvement obtained by the inconsistency resolution methods SAQ and MV.

We define the accuracy improvement Acc_{imp} as the difference between the obtained accuracy after inconsistency resolution and the max primary accuracy in the tools' set:

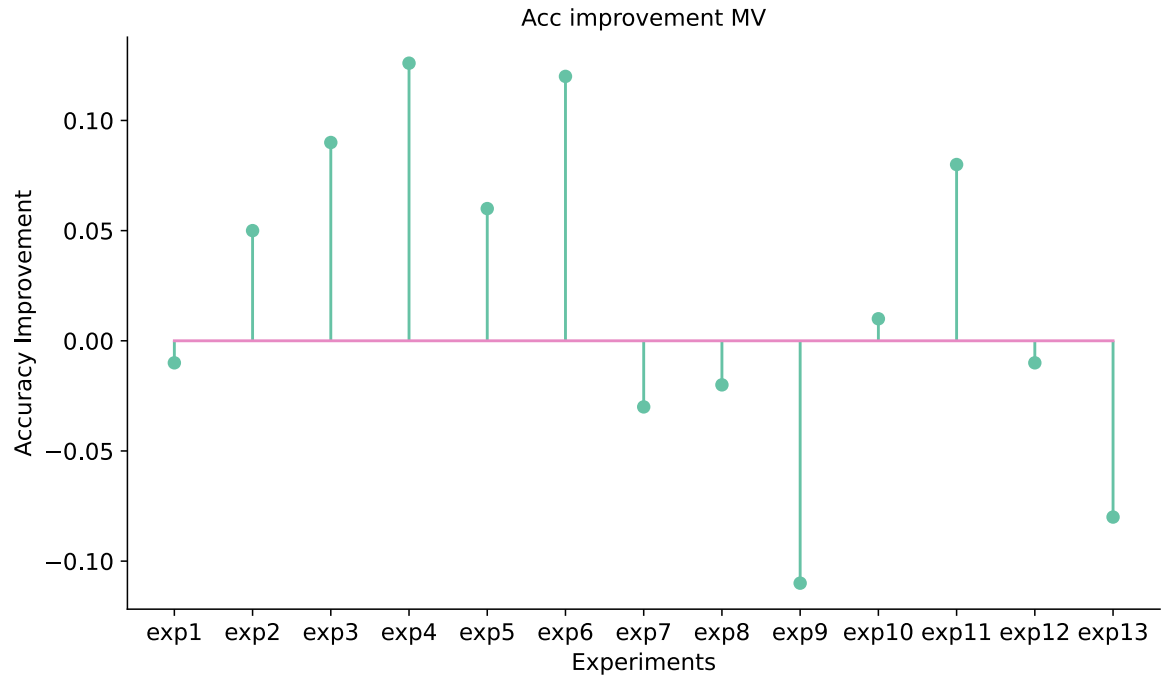
$$Acc_{imp} = Acc(inc\_res(\Gamma), D) - \max_{P_{t_k} \in \Gamma, D} Acc(P_{t_k}, D) \tag{4.4}$$

where P_{t_k} is a polarity function, Γ is the set of polarity functions, D is the dataset, and inc_res is an inconsistency resolution method. A positive value of Acc_{imp} means that the inconsistency resolution improves the accuracy, while a negative value means that accuracy has reduced after inconsistency resolution.

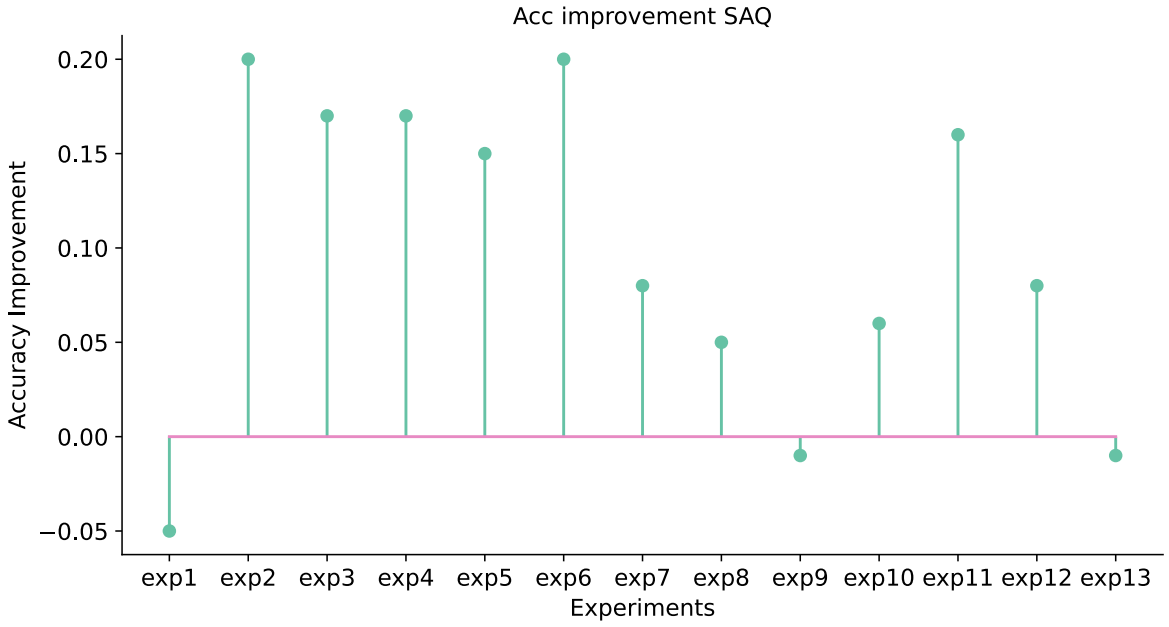
The accuracy values of tools in the used sets Γ in this experiment are presented in Table 4.3. Figure 4.3 presents the statistical summary of the sets. Figure 4.4a and Figure 4.4b display the experimental results of running SAQ and MV, respectively, on the sets of tools.

Every set $exp1$, $exp2$, $exp3$, $exp4$, and $exp5$ contains 4 polarity functions with the same accuracy values (0.3, 0.4, 0.5, 0.6, and 0.7, respectively).

We observe an accuracy improvement with the sets $exp2$, $exp3$, $exp4$, and $exp5$



(a) Acc_{imp} using MV



(b) Acc_{imp} using SAQ

Figure 4.4: Accuracy improvement in SAQ and MV comparing to tools primary accuracy

exps	$Acc(P_{t_1}, D)$	$Acc(P_{t_2}, D)$	$Acc(P_{t_3}, D)$	$Acc(P_{t_4}, D)$
$exp1$	0.3	0.3	0.3	0.3
$exp2$	0.4	0.4	0.4	0.4
$exp3$	0.5	0.5	0.5	0.5
$exp4$	0.6	0.60	0.6	0.6
$exp5$	0.7	0.7	0.7	0.7
$exp6$	0.61	0.55	0.57	0.6
$exp7$	0.46	0.59	0.52	0.56
$exp8$	0.76	0.65	0.51	0.88
$exp9$	0.56	0.55	0.51	0.88
$exp10$	0.6	0.6	0.6	0.4
$exp11$	0.6	0.6	0.4	0.4
$exp12$	0.6	0.6	0.4	0.4
$exp13$	0.6	0.4	0.4	0.4

Table 4.3: Accuracy of tools

when resolving inconsistency using MV and SAQ with a more significant improvement with SAQ. We obtain an accuracy degradation on the exp1 since the proportion of incorrect prediction is more considerable, leading to incorrect labels propagation and decreasing accuracy.

The set $exp6$ is a set of polarity functions with a mean accuracy of 0.58, a max accuracy of 0.61, and a min accuracy of 0.55. When resolving inconsistency on exp6, we obtain an accuracy improvement on both MV and SAQ with more significant improvement on SAQ (Acc_{imp} on SAQ is 0.2 and acc_{imp} on MV is 0.1).

The set $exp7$ is a set of polarity functions with divergent accuracy values ($\sigma = 0.14$, min accuracy of 0.46, and max accuracy of 0.59). In this experiment, we obtain an accuracy improvement on SAQ and an accuracy degradation on MV (acc_{imp} on SAQ is 0.07 and -0.03).

The sets $exp9$, $exp10$, $exp11$, $exp12$, and $exp13$ analyze the cases of sets with outlier accuracy values.

The sets $exp9$, $exp10$, and $exp13$ have max outliers, i.e., outlier polarity functions with the highest accuracy values.

We observe an accuracy degradation on exp9 (acc_{imp} -0.01 and -0.1 on SAQ and MV, respectively) and exp13 (acc_{imp} of -0.06 and -0.01 on MV and SAQ, respectively). At the same time, we notice a slight accuracy improvement on exp10 (improvement of 0.01 and 0.05 with SAQ and MV, respectively).

The set $exp11$ holds a lower outlier, and the inconsistency resolution leads to an accuracy improvement on the two methods with higher accuracy on SAQ.

On the set $exp12$, we obtain an accuracy improvement on SAQ and degradation on MV (negative improvement).

4.2.4 Accuracy Improvement and Number of Tools

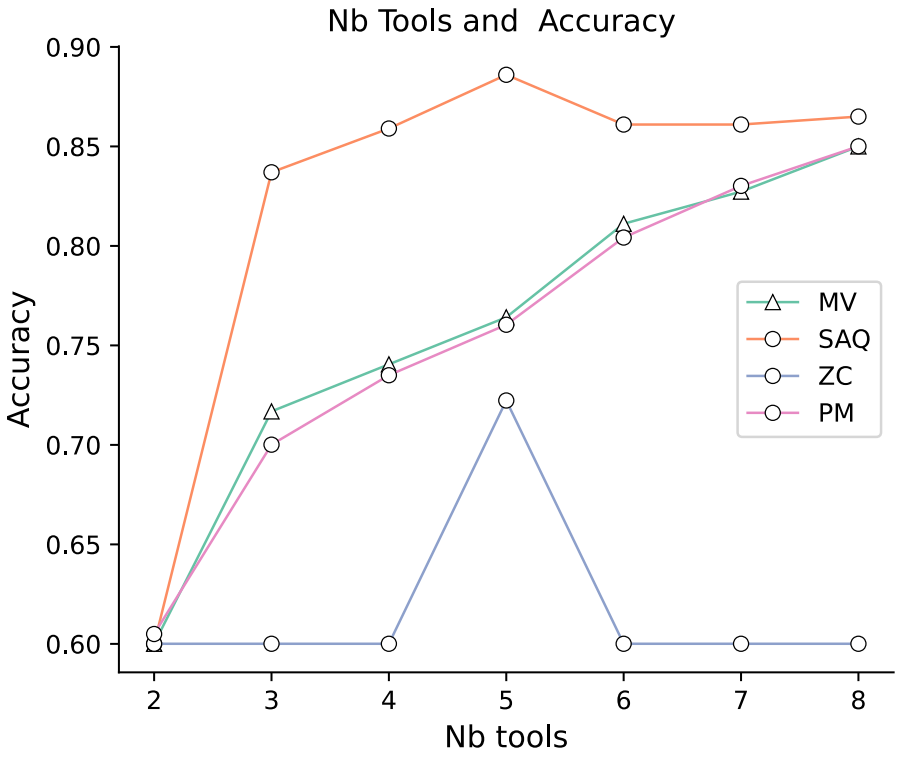


Figure 4.5: Accuracy improvement and the numbers of tools (Nb tools is the number of tools) after inconsistency resolution using the methods (SAQ, MV, ZC, PM)

This experiment evaluates the tools' set size impact on accuracy improvement obtained after inconsistency resolution. We analyze sets containing 2 to 8 tools such that every tool has an accuracy of 60%.

We resolve the inconsistency in each set using SAQ, MV, ZC, and PM methods and display the results in Figure 4.5.

We observe that when increasing the number of tools, the accuracy improvement increases on all methods except ZC.

We also observe that the accuracy improvement obtained by SAQ outperforms all other state-of-the-art methods and guarantees more significant improvement.

We conclude that the more we have tools, the more we obtain better accuracy improvement.

The efficiency of SAQ is significant when we have low redundancy of tools. The efficiency of SAQ is significant when we have low redundancy of tools. We observe, for instance, that with three tools, we have better accuracy improvement with SAQ comparing to MV, while with eight tools, the improvement difference between SAQ and MV is slight.

4.2.4.1 Discussion

This experiment studied the effect of tools' primary accuracy on the resulting accuracy improvement obtained after inconsistency resolution. The findings show that the primary accuracy of tools impacts accuracy improvement.

For instance, the case of tools' set with homogenous accuracy values (law values of standard deviation) and a significant proportion of valid predictions (high mean accuracy) guarantees an accuracy improvement after inconsistency resolution.

However, at the same time, the two cases of sets with a higher divergence between tools and outliers with max accuracy values lead to accuracy degradation when resolving inconsistency since we propagate invalid prediction.

We also notice that the case of tools' set with outliers having the lowest accuracy values leads to accuracy improvement with SAQ and accuracy degradation with MV.

Therefore, the usefulness of using SAQ, which uses a tools' ranking based on their inconsistency, is more apparent in the case of tools' set with outliers having the lowest accuracy values.

To illustrate this case, consider the Example 1.1. Applying the MV baseline on this example leads to attributing the polarity Negative to A3, which is not a correct polarity in this case. Because with simple majority voting, we consider only the inconsistency of polarity functions on the analogical set, and we ignore its behavior on the rest of the data. Hence, the usefulness of using inconsistency resolution methods that weight tools.

4.3 Learned Lessons

The experiments above show that resolving both inconsistencies is beneficial since SAQ, which resolves both inconsistencies, outperforms other inconsistency resolution methods that only resolve the inter-tool inconsistency.

The experiments reveal that we should use tools with homogeneous accuracy values (low standard deviation accuracy) and a significant proportion of valid labels (high accuracy mean) to guarantee performance improvement.

Majority voting is the trivial solution for the inconsistency problem. Nevertheless, this baseline considers only the voting subset and ignores the entire dataset's tools' behavior.

Hence the importance of using methods that wight tools based on their behavior on all data. The above experiments show that weighted tool methods significantly improve the accuracy in low tools redundancy.

In high redundancy, it is helpful to use majority voting, as mentioned in [2], [88].

4.4 Conclusion

This chapter presented SAQ, a framework for sentiment analysis quality that resolved both intra-tool and inter-tool inconsistency. It evaluated its performance compared to the state-of-the-art methods and studied the impact of resolving inconsistency on accuracy.

The initial results of SAQ are promising and show the efficiency of including semantics to resolve the intra-tool inconsistency to boost accuracy.

The findings presented in this chapter are helpful to improve inconsistency resolution methods in several domains, such as crowdsourcing and weak supervision.

Using logic to formalize the inconsistency resolution problem allows guiding and improving inconsistency resolution by adding business rules.

The promising primary results of SAQ motivate us to explore the use of inconsistency resolution to develop a weakly supervised method for sentiment analysis and test its efficiency on a real dataset.

In the next chapter, we propose a new semantic-based weakly supervised method for sentiment analysis that uses the aggregate output from different sentiment analysis and explores the semantic similarity between documents to infer new probabilistic labels.

Chapter 5

A Semantic Based Weakly Supervised Method for Sentiment Analysis

Based on the finding in Chapter 3 and Chapter 4, this chapter aims to introduce a novel sentiment analysis method WSSA (Semantic-based Weakly Supervised approach for Sentiment Analysis), based on the paradigm of weak supervision.

WSSA considers a set of sentiment analysis tools that provide noisy polarity labels (weak labels), then resolves inconsistencies respecting the consensus: semantically near documents have near polarity values, and each document has a unique polarity. The approach assigns weights to the tools and ranks them according to their consistencies. Then aggregates polarities based on this ranking and domain polarity knowledge using Probabilistic Soft Logic (PSL).

Since learning in PSL is supervised and needs to have a ground truth of polarities, we propose a novel unsupervised logic-based approach to learn tools' weights through logical entailment explanation. The resulting probabilistic labels are used as input of the voting model that selects the adequate polarities.

Moreover, we demonstrate the usefulness of our approach compared to the unsupervised learning methods from the state-of-the-art and the primary tools used as sources of weak labels.

To the best of our knowledge, this work is the first to handle at the same time the two types of inconsistency by considering labels from different weak sources and semantic similarity between documents with domain knowledge to perform weak supervision for sentiment analysis.

In the remainder of the chapter, we present in section 1 the motivations behind WSSA. Then, we present the intuition, an overview of the WSSA, and the different modeling steps in section 2. After that, in section 3, we present the experiments and discuss the results. Finally, we conclude and present further opportunities for future works.

5.1 Motivation

As presented in Chapter 4, traditional sentiment analysis tools such as lexicon-based methods and lexicon-based methods with rules layer are inadequate for polar facts. However, this limitation has been overcome using traditional machine learning methods that extract polarity based on sentiment features derived from the document.

Furthermore, the success of deep learning over traditional statistical models has reached sentiment analysis because of the ability to perceive learning features instead of manual feature engineering. For instance, the recent approaches [40], [47], [60], [67] use deep learning to learn the sentiment features and extract the polarity based on the learned features, achieving higher performances than traditional learning methods.

Deep learning approaches are robust and highly accurate when trained with a massive mass of high-quality labeled data. However, training such models is time-consuming, expensive, and requires that annotators have prior knowledge of the domain. Therefore, many studies adopt a crowdsourcing paradigm, a fast and less expensive alternative for generating training data. For example, some works use the user's reviews ratings as labels, while others use the emojis present in tweets as labels for the training data.

Nevertheless, despite the low cost of these methods, the labels they generate are of lower quality, which may harm the resulting models' quality and produce inconsistencies in predictions. Therefore, many research works have adopted the weak supervision paradigm that performs classification by considering that training labels are noisy.

Furthermore, based on our study in Chapter 3, many research works have considered inconsistency resolution to perform weak classification. However, to the best of our knowledge, no previous work has handled the two types of inconsistencies by considering labels from different sources and semantic similarity between documents to resolve the inconsistency.

The findings in Chapter 3 reveal that resolving the two types of inconsistency allows obtaining an accuracy improvement of 20% when considering a set of tools with 60% accuracy.

Therefore, the idea of WSSA a weakly-supervised approach for sentiment analysis, which we present in this chapter, is to consider noisy labels from different sources and to resolve the inconsistency based on consensus:

- The semantically close documents have close polarities, i.e., when the documents have a likely meaning, they have near polarity values.
- All tools should extract the same polarity for the document, i.e., all tools should agree on one polarity for the document. A document may have many polarities like the example (the device is suitable, but the screen has problems), where we have Positive and Negative polarities for different aspects. We consider only sentences with one aspect in our work, as presented in the Example 5.2.

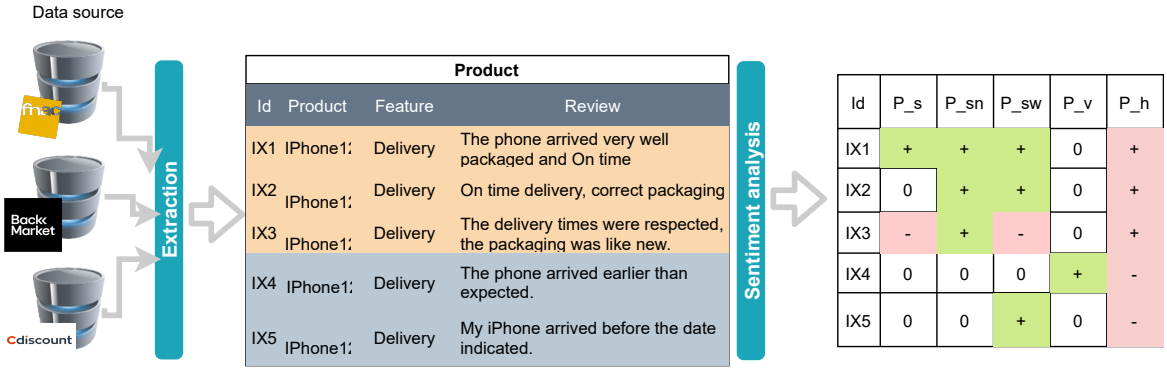


Figure 5.1: Running example

In the next section, we present a running example.

5.2 Running Example

We consider the real-life example of Figure 5.1, which represents customers' opinions collected from French e-commerce websites about the provided delivery services identified in $D = \{IX1, \ldots, IX5\}$, where each element of this dataset D is a *Document*. We notice that D can be clustered on subsets of semantically equivalent documents (analogical sets). For instance, $IX1$, $IX2$, and $IX3$ are semantically equivalent as they both express the idea that the delivery time was respected and that the packaging was correct. We denote this set by A_1 and we write: $A_1 = \{IX1, IX2, IX3\}$ and $A_2 = \{IX4, IX5\}$, which express that the phones arrived before time. We have: $D = A_1 \cup A_2$. We analyze D using four sentiment analysis tools: Stanford Sentiment Treebank [40], Sentiwordnet [52], senticnet [29], and Vader [8].

In the rest of this chapter, we refer to the results of these tools using the polarity functions: P_s, P_{sw}, P_{sn}, and P_v; we use P_h to refer to the ground truth. Figure 5.1 summarizes the results of the motivating example.

We know that each document has one polarity, so all tools' predictions should be equal, and a different polarity means that at least one tool is erroneous on this document. We also know that semantically equivalent documents should have the same polarity.

However, in this real-life example, we observe different tools attributing different polarities for the same document (e.g., only P_s attributes the polarity *"Negative"* to $IX3$ while P_sn have attributed a positive *"Polarity"* which represent an inter-tool inconsistency (Definition 18).

Also, the same tool attributes different polarities for semantically equivalent documents (e.g., P_s considers $IX1$ as positive and $IX3$ as Negative) which represents an intra-tool inconsistency (Definition 18).

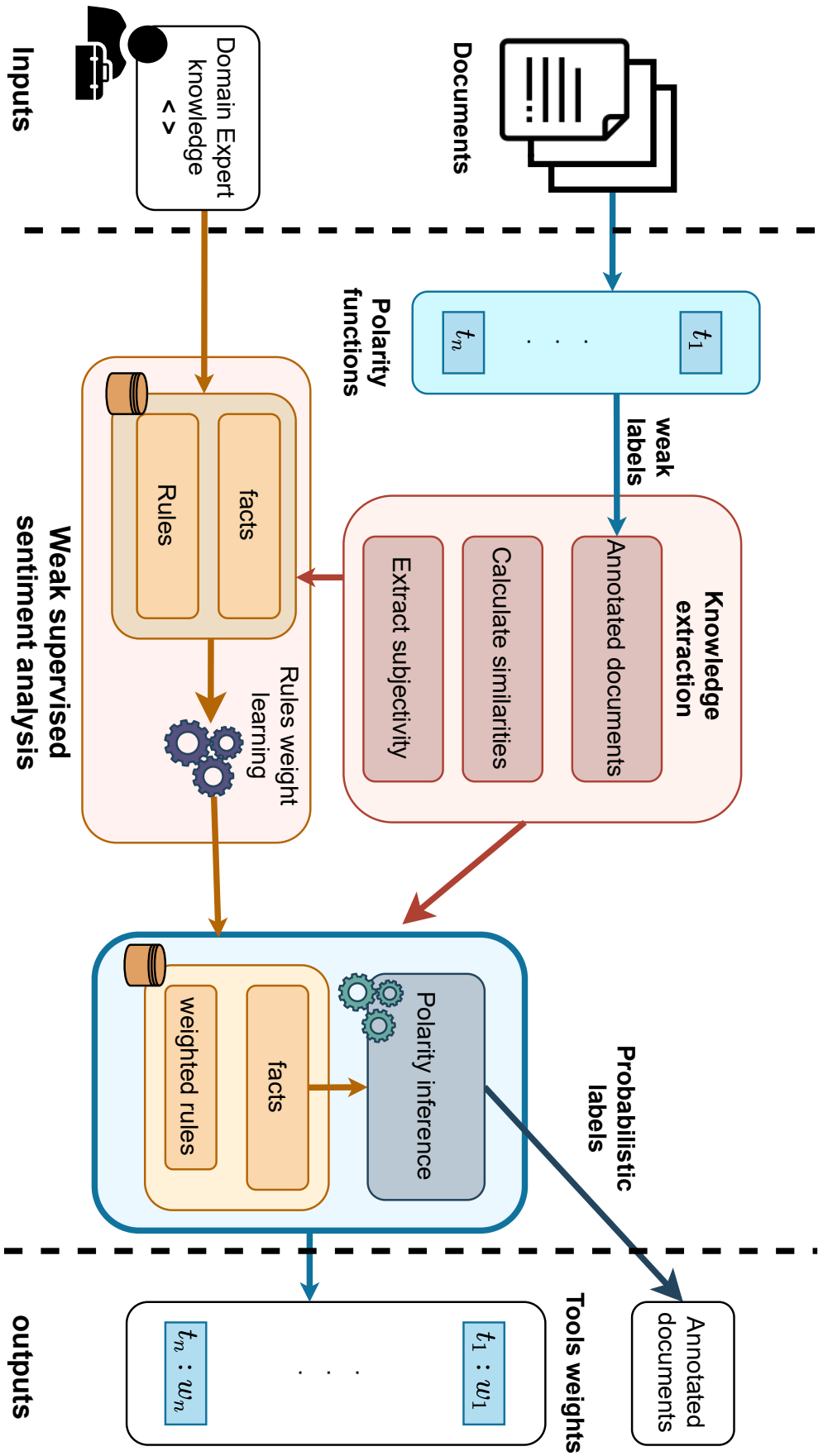


Figure 5.2: WSSA Overview

5.3 WSSA: Weakly Supervised Semantic Based Sentiment Analysis

In this section, we first present the overview of our weak supervised approach for sentiment analysis and the intuition behind it, and then we show the details of each system's component.

5.3.1 WSSA Overview

In this approach, we consider a dataset to be annotated with polarity labels. However, we face the problem that we do not have an annotated dataset to train a classifier for that task, and the existing tools are not enough accurate on that dataset.

Therefore, to extract polarities from that dataset with higher accuracy, we consider a set of sentiment analysis tools as a source of weak labels. Then we aggregate these polarities based on the tool's consistency and domain expert knowledge that aims to guide and help the reasoning.

WSSA is based on probabilistic logic modeling to integrate domain expert knowledge and handles the two types of inconsistency and uncertainty. It aggregates outcomes from different weak sources and associates each document with probabilistic labels.

The intuition behind our method is to resolve inconsistencies between tools to reduce the number of invalid predictions and converge to the golden truth. The use of business rules allows guide and help the convergence by providing additional knowledge. The overview of WSSA is depicted in Figure. 5.2.

We consider different features to aggregate the labels: objectivity of the document, semantic similarity between documents, the consistency of tools, and domain expert knowledge. We assume that if documents are semantically equivalent, they have close polarity values.

The strict assumption in Chapters 3 and 4 asserting that equivalent documents should have the same polarity is relaxed because of the soft calculation for the semantically equivalent relation between documents. Various works such as [3], [29] have adopted this relaxation.

Moreover, based on the findings in Chapter 4, which assert that tools are more accurate on subjective documents, we define a preference order between documents polarities based on their subjectivity.

In the next sub-section, we present in detail the following steps of WSSA:

1. We first sample a training dataset.
2. We construct the knowledge base $\mathcal{KB}$ by extracting weak labels using sentiment

analysis tools, calculating semantic similarities between documents, extracting the subjectivity properties, and integrating the expert's knowledge.

3. We learn the tool's weights based on their consistency.

4. We aggregate the different polarities using the PSL model to get probabilistic labels.

5.3.2 Knowledge Base Construction

The main component of WSSA is a knowledge base $\mathcal{KB}$ which comprises a fact set containing the data (instances) and a rules' set that defines the constraints, using specific vocabulary shown in Table 5.1 with its semantic and truthiness calculation. The vocabulary contains:

1. concepts: *Document* represents the text to evaluate, the subjectivity of the document represented by $isFact$ and $IsSubjective$, polarity functions P^{Ω}_{tk} that represent the sources of the weak labels

2. relations: $SameAs$ is the semantic similarities between documents, $Pref^{=}$ and $Pref^{>}$ are the preference relations between documents based on their subjectivity.

5.3.2.1 Fact base construction.

To construct the fact base, we need first to extract the different knowledge as follows:

Add documents. Each document d_i is represented by the concept $Document(d_i)$ and is associated with truth value 1.

Sentiment analysis. To construct the fact base, for each tool t_k, and each document d_i, we add the concept $P^{\Omega}_{t_k}(d_i)$ with Ω the polarity given by t_k to d_i. The truth value is 1 if the polarity given by t_k to d_i is Ω, 0 otherwise.

Semantic similarity. To express the $SameAs$ relations between documents, we calculate the $Word2Vec$ embedding [70] of each word in the document, then we calculate the average. Formally speaking, let us consider the document d_i as a set of n words such that

$$d_i = \{x_{i1}, ..., x_{in}\}$$

We apply word embedding to each word of the document. We get a matrix representation $M^{n \times m}$ for this document such that $M_i = f_{emb}(x_i)$ and f is the embedding of the document.

After that, we average the matrix columns to get a vector X, the document's representation is given by:

$$X_{d_i} = \frac{1}{n} \sum_{i \in n} f_{emb}(x_i)$$

Then, we calculate the COS semantic similarity $Cos\_sim(X_{d_i}, X_{d_j})$ between each two documents d_i and d_j

$$Cos\_sim(X_{d_i}, X_{d_j}) = \frac{X_{d_i}., X_{d_j}}{\|, X_{d_j}\|.\|X_{d_i}\|}$$

This similarity score represents the truth value of $SameAs(d_i, d_j)$.

Subjectivity We calculate the document's subjectivity $sc(d_i)$ using a pattern-based method that takes word subjectivity scores from Sentiwordnet [41] and averages them.

We determine the truth values of the concepts $IsFact(d_i)$ and $IsSubjective(d_i)$ based on the subjectivity score of the document $sc(d_i)$ such that $1 - sc(d_i)$ is the truth value of $IsFact(d_i)$ and $sc(d_i)$ is the truth value of $IsSubjective(d_i)$.

Example 5.3.1.

Let's consider the running example in Section 5.2. After calculating the subjectivity score of each review, we found the following subjectivity scores sc :

$sc(IX1) = 0.5$, $sc(IX2) = 0$, $sc(IX3) = 0.65$, $sc(IX4) = 0.6$, and $sc(IX5) = 0$.

These scores lead to deriving the following subjectivity for the documents a.k.a reviews: $IsSubjective(IX1)$, $IsFact(IX2)$, $IsSubjective(IX3)$, $IsFact(IX4)$, and $IsSubjective(IX5)$.

5.3.2.2 Rules

In addition to the concepts, the rules' base $\mathcal{R}$ contains a set of tools consistency rules, a set of preference rules between documents, and a set of domain experts' knowledge rules, as well as a set of rules related to polarity disjunction.

Inter-tool consistency rules These rules model the inter-tool consistency presented in Equation(3.3) and assert that each document has a single polarity and this polarity is given correctly by the tool:

$$IR1 : isPositive(d_1) \leftarrow P^{+}_{t_k}(d_1)$$
$$IR2 : isNegative(d_1) \leftarrow P^{-}_{t_k}(d_1)$$
$$IR3 : isNeutral(d_1) \leftarrow P^{0}_{t_k}(d_1)$$

For each used tool, we create the following rules by replacing P_{t_k} with the polarity function of the tool.

Predicates	Semantics	Truth values	Truth Threshold
$Document(d_1)$	It models the document to be annotated	1.0 for each document to be annotated	-
$DomainConcept(ex_i)$	It models list of polarity words/concepts/ expressions given by domain experts	1.0 for each given expression	-
$P^{\Omega}_{t_k}(d_1)$	The polarity function of the tool that models the polarity Ω assigned by the tool t_k to the document d_1 with $\Omega \in \{Positive, Negative, Neutral\}$.	1.0 if the tool t_k assigns the polarity Ω to the document d_1, else 0.0	-
$isPositive(d_1)$ $isNegtive(d_1)$ $isNeutral(d_1)$	It represents the polarity of the document.	Inferred	-
$SameAS(d_1, d_2)$	It models the semantic similarity between documents d_1 and d_2	The cos similarity between d_1 and d_2	if $cos\_sim(d_1, d_2) > 0.8$ $sameAs(d_1, d_2) = True$ $elseFalse$
$Pref^{>}(d_1, d_2)$	It indicates that we trust the polarity of d_1 more than the polarity of d_2	Infered	-
$Pref^{=}(d_1, d_2)$	It indicates that we have no preference between d_1 and d_2 polarities	Infered	-
$IsFact(d_1)$	It asserts that the d_1 document is a polar fact	$1 - sc(d_i)$	if $sc(d_1) < 0.6$ $IsFact(d_1) = True$ else $False$
$IsSubjective(d_1)$	It asserts that the d_1 document is a opinionated	$sc(d_i)$	if $sc(d_1) >= 0.6$ $IsSubjective(d_1) = False$ else $True$
$Contains(ex_i, d_1)$	It asserts that the document d_1 contains the expression ex_i	1 if the document contains w_i	-

Table 5.1: List of concepts and relations, their semantics, the sources of the truth values, and the truth threshold

Preference rules Based on the subjectivity of the document, we define two preference orders between documents based on the trust for the document's polarity given by the tool.

For instance, if we have an opinionated document similar to a document that is a fact, we trust the polarity given by the tool to the opinionated document more than the polarity given to the fact, since tools are more accurate on subjective documents than facts (see sub-section 3.3.5.8). We write:

$$P1 : Pref^{=}(d_1, d_2) \leftarrow SameAs(d_1, d_2) \wedge Subjective(d_1) \wedge Subjective(d2)$$
$$P2 : Pref^{>}(d_1, d_2) \leftarrow SameAs(d_1, d_2) \wedge Fact(d_1) \wedge Subjective(d_2)$$
$$P3 : Pref^{=}(d_1, d_2) \leftarrow SameAs(d_1, d_2) \wedge Fact(d_1) \wedge Fact(d_2)$$

If both documents, d_1 and d_2, are facts or both are subjective, we trust their polarities to the same degree; therefore, we have no preference order in this case.

However, if one document is subjective, and the other is a fact, we trust more the polarity of the subjective document since tools are more accurate on subjective documents than facts.

Intra-tool consistency rules based on the preference These rules assess the intra-tool consistency by assuming that the more documents are semantically close, the more they have close polarities.

Based on the preference between documents, we define the intra-tool consistency rules. The preference order is related to symmetry, $'='$ preference ($Pref^{=}$) is symmetric, but $'>'$ preference ($Pref^{>}$) is not.

We assume that if a document is a fact and is close to a document that is subjective, we guess that the fact document should get the polarity of the subjective document.

$$PI1 : IsPositive(d_2) \leftarrow IsPositive(d_1) \wedge Pref^{=}(d_1, d_2)$$
$$PI2 : IsNegative(d_2) \leftarrow IsNegative(d_1) \wedge Pref^{=}(d_1, d_2)$$
$$PI3 : IsNeutral(d_2) \leftarrow IsNeutral(d_1) \wedge Pref^{=}(d_1, d_2)$$
$$PI4 : IsPositive(d_2) \leftarrow IsPositive(d_1) \wedge Pref^{>}(d_1, d_2)$$
$$PI5 : IsNegative(d_2) \leftarrow IsNegative(d_1) \wedge Pref^{>}(d_1, d_2)$$
$$PI6 : IsNeutral(d_2) \leftarrow IsNeutral(d_1) \wedge Pref^{>}(d_1, d_2)$$

Example 5.3.2.

> Following the running example in Section 5.2, IX1 is subjective, IX2 is a fact, and IX1 and IX2 are semantically equivalent. Rule P2 allows setting a preference order between IX1 and IX2; then, we trust the polarity given by tools to IX1 more than the polarity of IX2.

Disjunction rules They are considered hard rules that can not be violated, and they assume each document has a unique polarity.

Since the rules in our approach are of the form $H \leftarrow \bigwedge_{i=0}^{n} B_i$, we use the following rules to express the disjunction:

$$D1 : \neg IsPositive(d_1) \leftarrow IsNegative(d_1)$$
$$D2 : \neg IsNeutral(d_1) \leftarrow IsNegative(d1)$$
$$D3 : \neg IsNeutral(d_1) \leftarrow IsPositive(d1)$$

Domain expert's rules These rules are given by domain experts and help to learn tools weights and aggregate different polarities. To guide the expert, we provide the following patterns for business rules:

$$DT1 : \Omega(d_j) \leftarrow \bigwedge_{i=0}^{n} contains(ex_i, d_j)$$
$$DT2 : \Omega(d_j) \leftarrow P_h^{\Omega}(d_i)$$

where d_j is a document, P_h^{Ω} the golden truth or the expert annotation for the document d_j and $\Omega(d_j) \in \{isPositive(d_j), isNegative(d_j), isNeutral(d_j)\}$, ex_i domain concepts for polarity.

The first pattern $DT1$ states that if a document contains an expression (word concept or text), then it has the polarity Ω. For that, we have defined the relation $contains(ex_i, d_j)$. To define the truth value of this relation, we search for the exact expression in the documents, if it is found we assign the truth value 1 to that concept. Otherwise, we do not ground the concept to reduce the reasoning time.

The second patterns represent a set of documents annotated by experts. It asserts that if an expert annotates the document with polarity Ω, this polarity is the true polarity of the document. Note that these rules are hard.

Through the *SameAs* relations and the overall reasoning, these rules help to reduce the number of incorrect labels and guide the reasoning.

Example 5.3.3.

Based on the running example of Section 5.2. We define the following rules of type DT1 and DT2:

$IsPositive(d_j) \leftarrow contains(\text{"}correct\,packaging\text{"}, d_j)$

$IsPositive(d_j) \leftarrow contains(\text{"}Ontime\text{"}, d_j) \wedge contains(\text{"}delivery\text{"}, d_j)$

$IsNegative(IX5) \leftarrow P_h^{+}(IX5)$

Based on these rules, we consider each review that contains the expression "correct packaging" as Positive, and the presence of the expressions "On time" and "delivery" on the same review indicates that this review is positive.

The third rule asserts that the review IX5 is Negative, then each review similar

to IX5 is considered Negative. Hence, the importance of domain expert rules that allows guiding the reasoning.

5.3.3 Weights' Learning

Many research works have proposed algorithms for rules weight learning in PSL such as [106], [127], [128]. Typical methods such as [106], [128] learn the weights of the rules by maximizing a likelihood function instead of the method in [127] that finds a Bayesian optimization for weights based on given domain metrics.

However, all these research efforts require observed data (golden truth) for the optimization, which is not proper for our case since we do not have valid labels or have only a tiny set of correct labeling given by business rules of type $DT2$. Hence, we propose in our approach to learn the weights based on the consistency of tools represented by the agreement and disagreement between inter tool consistency rules, preference between data, and expert knowledge.

We define the two notions of agreement and disagreement between rules as follows:

Definition 20. ***(Agreement)***. *Two rules R_i and R_j are in an agreement iff: $R_i \models a$ and $R_j \models a$, where a is an entailment.*

Definition 21. ***(Disagreement)*** *Two rules R_i and R_j are in a disagreement iff $R_i \models \neg a$ and $R_j \models a$, where a is an entailment.*

Knowing the number of agreements and disagreements between rules allows to know the number of rules that contradict a rule R_i (inconsistent rules) and the number of rules that support its conclusion. The pseudo algorithm of our learning approach is presented in Algorithm 8.

To calculate the number of agreements and disagreements of a rule's grounding, we calculate the number of logical explanations of the rule's entitlement α and the inverse of the rule's entailment $\neg\alpha$, respectively.

Many algorithms exist to find all logical explanations, a.k.a justifications, that come in two flavors: black-box algorithms implemented on the top of a reasoner and glass box built on the existing tableau-based procedure for expressive Description Logic (DL) language.

For that, we choose the procedure in [129] that uses the SEARCH HST algorithm to calculate all justification in OWL-DL $\mathcal{SHOIN}^{(D)}$.

5.3.3.1 Knowledge base construction

The first step of the algorithm is to sample a training batch D' from the dataset D, then we construct the learning knowledge base that corresponds to the sampled

Algorithm 8 Calculate_weights

Input :
R: rules list; D: set of documents; P: weak labels, nbepochs: number of epochs
Output : W: rules weights)

procedure
  for nbepochs
    $D' = sample(D, size)$
    //**Step1:** Build knowledge base
    $\mathcal{KB} = construct\_kb(R, D', P)$
    for each $r_i \in R$:
      $RG_i = \text{ground}(r_i)$ (*Algorithm 9*)
      for each $rg_{ij} \in RG_i$:
        get α s.t $rg_{ij} \models \alpha$
    //**Step2:** Calculate the number of agreements
        $n = explain(kb, R/r_i, \alpha)$
    //**Step3:** Calculate the number of desagreements
        $m = explain(kb, R/r_i, \neg\alpha)$
        $N.add(n)$
        $M.add(m)$
    //**Step4:** Calculate weight w_i *(Equation 5.1)*
      $w_i = calculate\_weights(M, N)$
      $W.add(w_i)$
    return W

batch.

Since many concepts are soft, which do not fit with $\mathcal{SHOIN}^{(D)}$ fragment where the truth values are binary, we fix thresholds to convert the soft truth values into binary values. The thresholds are presented in TABLE 5.1. Note that the thresholds were determined empirically.

The expressivity of our modeling fragment is included in $\mathcal{SHOIN}^{(D)}$ since in our modeling, we use concepts, concepts' negation, roles, and concept inclusion. This modeling is supported by $\mathcal{SR}$, a sub-fragment of $\mathcal{SHOIN}^{(D)}$.

5.3.3.2 Grounding

We ground each soft rule using our proposed grounding algorithm; its pseudo-code is described in Algorithm 9. Our grounding algorithm is inspired by the work [4], where they consider the $\mathcal{KB}$ as a relational database DB and build a grounding SQL query Q based on the rule.

The novelty of Algorithm 9 is to fully entails the implicit knowledge then grounds rules based on the fully entailed KB. It reduces the number of the generated grounded rules by avoiding considering groundings that do not exist rather than the algorithms that generate all the rule's grounding (even those that do not exist). To illustrate this, let us consider the following example:

Example 5.3.4.

We consider the two reviews $IX1$ and $IX2$ with the tool P_s from the running example of Section 5.2. The grounding of the rules is run on the following set that represents the observed data:

$$\begin{aligned} DB = \{&Document(IX1), Document(IX2), P_s^+(IX1), \\ &P_s^0(IX2), IsSubjective(IX1), IsFact(IX2)\} \end{aligned}$$

The grounding process by applying algorithm in [4] on this observed data will generate 15 grounded rules: one rule grounding for $IR1$, one rule grounding for $IR3$, and one rule grounding for $P2$. Since this process generates all possible groundings for the not observed concepts, and the concepts $IsPositive(d_i)$, $IsNegative(d_i)$, $IsNeutral(d_i)$, $Pref^{>}(d_i, d_j)$, and $Pref^{=}(d_i, d_j)$ are not observed, the grounding process generates two grounding for each rule $IP1$ to $IP6$.

However, our proposed grounding algorithm (Algorithm 9) fully entails the DB, generating DB' such that:

$$\begin{aligned} DB' = \{&Document(IX1), Document(IX2), P_s^+(IX1), P_s^0(IX2), \\ &IsSubjective(IX1), IsFact(IX2), Neutral(IX2), Positive(IX1), \\ &IsPositive(IX2), Pref^{>}(IX1, IX2)\} \end{aligned}$$

We ground only the values in the set. We obtain 4 rules grounding: one rule grounding for $IR1$, one rule grounding for $IR3$, one rule grounding for $P2$, and one rule grounding for $PI4$. Hence, this algorithm is useful to reduce the number of unnecessary groundings.

For instance, the number of resulted ground rules on SST for learning with our procedure is 4475 comparing to 110280 ground rules on the inference using the lazy grounding procedure.

5.3.3.3 Compute the number of agreement and disagreements

We compute the number of agreements and disagreements n and m respectively of each rule's grounding rg_{ij} by counting the explanations of α and $\neg\alpha$ respectively, with $rg_{ij} \models \alpha$ in the knowledge base $\mathcal{KB}$ and $R - \{r_i\}$.

We repeat this process for all epochs.

Algorithm 9 FE_grounding

Input : R, DB
Output : Full entailed DB

```
1: procedure
2:       //Step1: create and fill database
3:          //with prior knowledge
4:       for each Predicate p: create_table(p)
5:        while¬stop :
6:        stop = true
7:        for each r_i ∈ R :
8:          //Step2: build query to ground the body
9:           Q = build_query(body(r_i))
10:          L = execute(DB, Q)
11:         //Step3: based on the body grounding, entail
12:            //the rules conclusion and insert it if
13:            //it does not exist in the head's table
14:         //Step4: stop if no new lateral
15:            //inserted, else continue
16:           stop = false
```

5.3.3.4 Calculating Weight

We calculate the rule's weight which is the aggregation of agreements and disagreements of the rule. The rule with a higher agreement score and lower disagreement score is the rule that has the higher weight.

Hence, we consider the following objective O to aggregate the agreements and the disagreements:

$$O = \gamma \sum_j w_j \log w_j m_j - \beta \sum_j w_j \log w_j n_j$$

where $\gamma \sum_j w_j \log w_j m_j$ represents the penalty of the rule to minimize and $\beta \sum_j w_j \log w_j n_j$ represents the profit to maximize, γ and β are respectively regulation parameters. We calculate w_i that minimizes the objective such that

$$\underset{w_i}{argmin}\, \gamma \sum_j w_j \log w_j m_j - \beta \sum_j w_j \log w_j n_j$$

This objective is convex. Therefore, to resolve the optimization problem, we calculate the derivative of the objection on w_i. We found after simplification:

$$w_i = \sqrt[\gamma+\beta]{\frac{e^{-\gamma n_i - \beta m_i}}{n_i^{\gamma} m_i^{\beta}}} \tag{5.1}$$

5.3.4 Polarity Inference Process

The purpose of the PSL inference in our approach is to fusion and aggregate the different polarities provided by: the weak tools, domain expert knowledge, preferences, and semantic similarity between data represented by the rules defined previously.

We use the learned weights in the previous step as weights for the PSL soft rules. Hence, the important rules have higher weights which correspond to rules with lower disagreement and higher agreement.

We apply MPE inference (subsection 2.4.2.1) to infer the probabilistic polarities represented by vector $V_{p_*(d_i)} = [p_*^+(d_i), p_*^0(d_i), p_*^-(d_i)]$ where $p_*^+(d_i)$ is the probability that the document is Positive, $p_*^0(d_i)$ is the probability that the document is Neutral, $p_*^-(d_i)$ is the probability that the document is Negative.

5.4 Experiments

This section evaluates our approach proposal, compares it to different state-of-the-art methods for inconsistency resolution, and demonstrates its usefulness in improving accuracy. Next, we present our experimental setup and further experimentation.

Statistics	# elements	# Clusters	# Positive	# Neutral	# Negative
Amazon	21483	10704	16944	2028	2511
News	1580	831	890	36	654
SST	3504	1033	1492	628	1384
Total	92995	34760	28752	15804	48439

Table 5.2: Statistics of used datasets.

5.4.1 Experimental Setup

5.4.1.1 Datasets

We use our benchmark presented in Chapter 3 to evaluate our approach for weak supervision. This benchmark contains semantically equivalent documented augmented with paraphrases. We recall statistics about the benchmark are presented in TABLE 5.2.

We use the augmented datasets in the evaluation and the original datasets to train machine learning algorithms. Our choice is justified by the different domains of the datasets, which allows us to evaluate our approach on different domains.

5.4.1.2 Tools

To generate weak labels, we used the following tools: *Sentiwordnet [41]*, a lexicon based method for sentiment analysis that is very used and popular; *Senticnet [29]*, a concept lexicon for sentiment analysis that uses a layer of rules to enhance sentiment analysis; and *Vader [8]*, a lexicon for sentiment analysis created by domain experts.

We also used *rec_nn [40]*, that learns the word embedding; *Charcnn [67]*, and *Textcnn [44]*, deep learning methods that use pre-trained word embedding. Note that these deep learning methods are trained on other datasets than the datasets used in our experiments.

We justify this choice by the availability of those tools, their different nature and domains, and their very use in literature.

5.4.1.3 Metrics

We evaluate *Accuracy* by calculating the rate of correctly predicted polarities compared to the golden polarity (human annotation) as follows:

$$Accuracy = \frac{\sum_{i=0}^{n} \mathbb{1}_{(P_h(d_i)=P_*(d_i))}}{n}$$

Since labels are probabilistic, we fix a probability threshold to choose the label.

5.4.1.4 Baseline

To evaluate the performance of our method in resolving inconsistency, we have compared it to the state-of-the-art methods for truth inference that resolves the inconsistency between workers.

Furthermore, since truth inference methods are classified to probabilistic model, direct computing, and optimization [88], we compare our method to the following baselines ZC, PM, and MV. More details about those methods are presented in Section 4.2.1.3.

5.4.1.5 Implementation details

Our tool and experiments were implemented in python and conducted on a server with 75GB of RAM and two Intel Xeon E5-2650 v4 2.2GHz CPUs.

We used the TextBlob[1] library to calculate the subjectivity; We used python implementation of PSL[2]; pellet reasoner[3] to perform logical explanation for weight learning. We implement a comprehensive grounding algorithm in python inspired from [4]. Our code and data are available.

5.4.2 Overall Performances

This experiment aims to show the efficiency of WSSA in resolving inconsistency, aggregate the labels, and improve accuracy.

For that, we compare the efficiency of WSSA to different state-of-the-art methods for inconsistency resolution from the truth inference domain: majority voting (MV), the probabilistic method ZC, and the optimization method PM.

We run the truth inference methods, and WSSA on the set of sentiment analysis tools in Section 5.4.1.2 applied on the datasets: news_headlines, sst, and amazon_ereviews.

The deep learning tools used on the news headline dataset were trained on the amazon_reviews dataset, while the tools used for SST and amazon reviews were trained on the news headline dataset.

We use the probability threshold of 0.6 on WSSA to choose one label. We plot the results in Figure 5.3.

We observe that our method outperforms the state-of-the-art methods in inconsistency resolution on the three datasets. This confirms that considering the semantic

[1]https://textblob.readthedocs.io/en/dev/
[2]https://psl.linqs.org/
[3]https://github.com/stardog-union/pellet

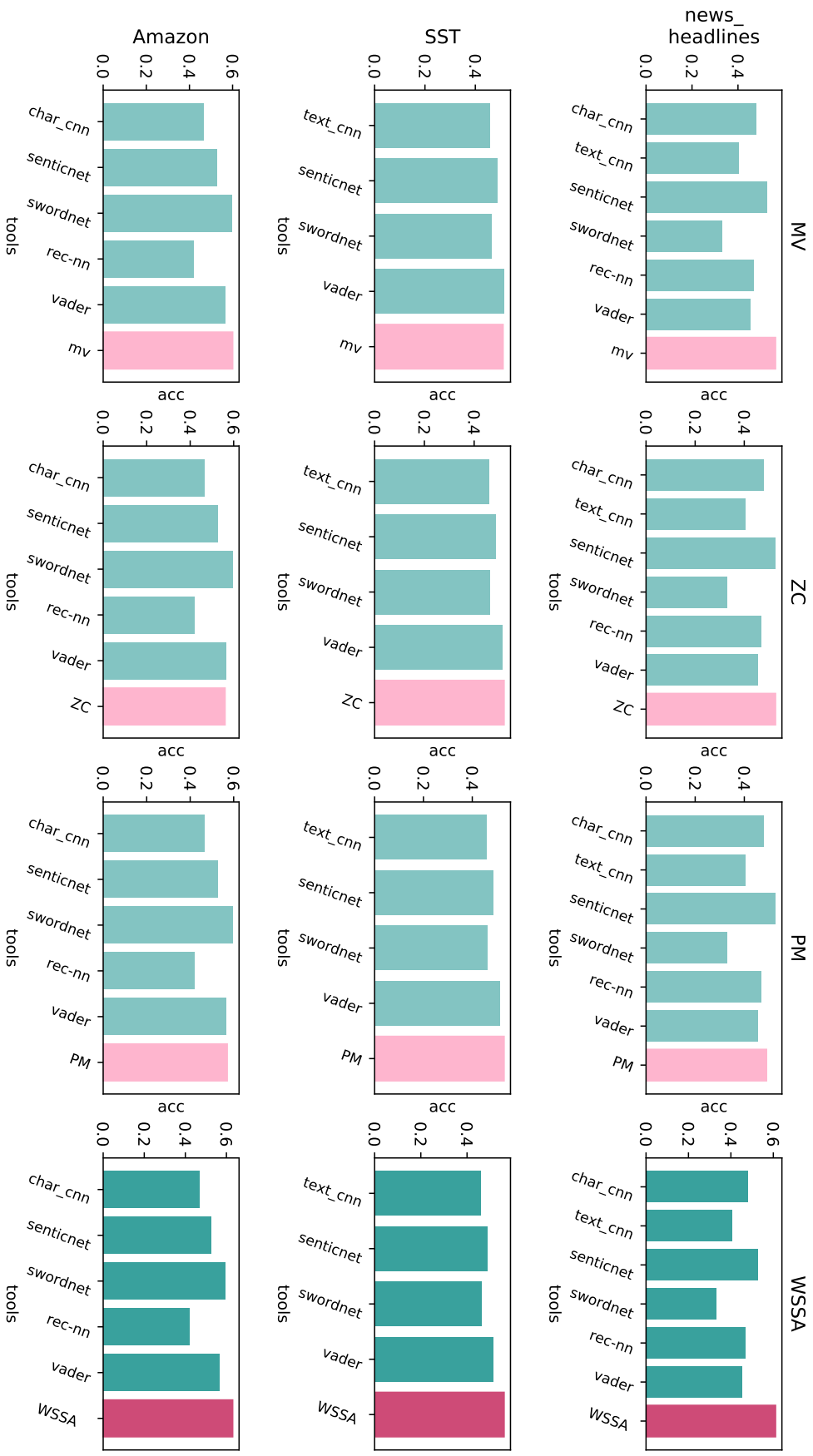


Figure 5.3: Performance of the WSSA comparing to state-of-the-art approaches (MV, ZC, and PM)

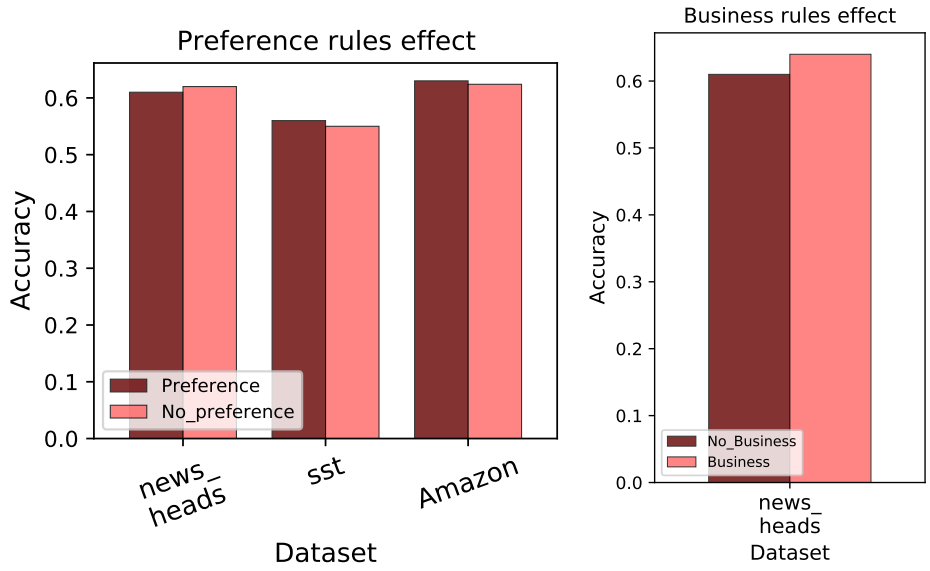


Figure 5.4: Rules Impact (**Left:** preference rules impact on accuracy, **Right:** domain expert rules impact on the accuracy).

similarity in inconsistency resolution improves accuracy better than only resolving inconsistency between tools.

5.4.3 Rules' Efficiency Evaluation

In this experiment, we evaluate the usefulness of each rule type on our approach. First, we run the three datasets news headlines, amazon reviews, and sst with preference rules, then with intra-tool consistency rules (see Section 4.1.2.2).

We evaluate the usefulness of business rules on the news headlines dataset. We use the set of rules presented in TABLE 5.3. The first group of rules (DT1) are particular polar expressions to the financial domain. The second group represents documents annotated with golden truth. We show results in Figure 5.4. We observe that the accuracy improvement obtained when using preference rules is higher on the SST and amazon review dataset than when we do not use preference rules.

We obtain lower accuracy on the news headlines dataset when using preference rules. We may explain this by the lack of subjective data on this dataset. However, using preference rules is helpful in SST and amazon reviews due to the nature of the data. The improvement may look slight, but this improvement is significant on big datasets like the amazon reviews dataset.

We observe that using business rules helps to improve the accuracy since they guide and control the inconsistency resolutions.

Pattern	rule
DT1	$isPositive(\text{"}r_1\text{"}) \leftarrow p_h^+(\text{"}r_1\text{"})$
	$isPositive(\text{"}r_2\text{"}) \leftarrow p_h^+(\text{"}r_2\text{"})$
	$isPositive(\text{"}r_3\text{"}) \leftarrow p_h^+(\text{"}r_3\text{"})$
	$isNegative(\text{"}r_4\text{"}) \leftarrow p_h^-(\text{"}r_4\text{"})$
	$isNegative(\text{"}r_5\text{"}) \leftarrow p_h^-(\text{"}r_5\text{"})$
	$isNegative(\text{"}r_6\text{"}) \leftarrow p_h^-(\text{"}r_6\text{"})$
DT2	$isPositive(d_1) \leftarrow contains(d_1, \text{"}sales\,rize\text{"})$
	$isNegative(d_1) \leftarrow contains(d_1, \text{"}eats\,into\,profit\text{"})$
	$isNegative(d_1) \leftarrow contains(d_1, \text{"}job\,cuts\text{"})$
	$isPositive(d_1) \leftarrow contains(d_1, \text{"}position\,opening\text{"})$
	$isPositive(d_1) \leftarrow contains(d_1, \text{"}hire\text{"})$

Table 5.3: Domain expert rules example

5.5 Conclusion

This chapter presents WSSA, a weakly supervised approach for sentiment analysis that considers several sources for weak labels then aggregates them.

WSSA uses probabilistic soft logic modeling to integrate the labels from different tools and handles inconsistency and uncertainty.

The originality of WSSA is to consider different features in the aggregation represented by the semantic similarity between documents, the subjectivity of the document by defining preference order between documents, and domain expert knowledge that defines the context and guides the reasoning. Moreover, this chapter proposes a novel unsupervised approach to learn rules weights for PSL. It is a new method that utilizes the logical explanation to compute the agreement and disagreement between tools.

Furthermore, this chapter presents a novel grounding algorithm that reduces the number of groundings by ground rules only with necessary grounding for the reasoning. The evaluation of WSSA proved that it outperforms different state-of-the-art methods for inconsistency resolution and the accuracy improvement demonstrates its usefulness.

WSSA combines symbolic AI and statistical AI-ML to improve the accuracy of sentiment analysis tools. Besides the outcomes of existing sentiment analysis tools, additional knowledge related to polarities and expert domains are used.

The obtained results open several research doors that we will present in the next section.

Chapter 6

Conclusion and Future Work

6.1 Summary

This thesis addresses the challenges related to sentiment analysis tools quality on the two levels, intra-tool, and inter-tool inconsistency. It studied first the inconsistency of tools and revealed the causes of the inconsistency, then studied the effect of the inconsistency resolution on sentiment analysis tools performances. After that, based on the study's findings, it proposes an elegant method for sentiment analysis based on the paradigm of weak supervision that aggregates outcomes by minimizing inconsistency.

At first glance, we have presented the first study that evaluates inconsistencies in sentiment analysis tools. The study took three dimensions: statistical analysis to qualify the inconsistencies on different sentiment analysis tools and datasets, structural analysis to evaluate the relationship between the document's structure and the inconsistencies, and semantic analysis to evaluate the relationship between document subjectivity and the inconsistencies. After that, we have evaluated the relationship between different hyper-parameters in deep learning and inconsistencies, then the relationship between inconsistency and accuracy. We have proposed a benchmark of paraphrases labeled with sentiment labels the allows the evaluation of the two types of inconsistencies and made it publicly available for further research. In the end, we have provided a set of recommendations to select a tool under a given scenario.

Based on the results of the inconsistency study, we have noticed that resolving inconsistency improves the accuracy when using majority voting, hence our study on the effect of resolving inconsistency on tool's performances. We have proposed SAQ, which allows resolving the two types of inconsistency: intra-tool and inter-tool inconsistencies by modeling the problem using Markov logic networks. We have evaluated the performances of SAQ in resolving inconsistencies and compare it to the state-of-the-art methods for inconsistency resolution (truth inference methods). Finally, we have studied the impact of resolving inconsistency of the accuracy and define the conditions that guarantee an accuracy improvement. The promising results of SAQ, particularly the accuracy improvement obtained of 20%, motivate us to extend it into a weakly

supervised method for sentiment analysis.

We proposed WSSA, a weakly supervised method for sentiment analysis that considers outputs from different weak sources then aggregates the different labels by reducing the inter-tool inconsistency and the intra-tool inconsistency between the documents. Moreover, it considers context knowledge represented by domain expert rules and sets a preference order between documents based on the subjectivity of the document.

In addition, WSSA uses a new grounding algorithm that allows generating only proper grounding, which reduces the number of generated grounded rules. A new semantic-based learning algorithm is proposed to learn rules' weights. The learning algorithm uses the logical explanation to rank rules based on agreement and disagreement. Our experiments proved that WSSA has integrated labels from different sources outperforming the state-of-the-art method for inconsistency resolution and the importance of different rules modeling.

6.2 Future Work

Our work opens up several research directions that we present below:

- Since our inconsistency resolution frameworks outperform truth inference methods in inconsistency resolution, the idea of considering outputs from different sources and resolving intra-tool and inter-tool inconsistencies to aggregate labels can be extended to be used in other contexts. For instance, it can be applied to truth inference in crowdsourcing [88], [130] to assess the workers' quality and infers the truth on different tasks on text data. Exploiting resolving intra-tool inconsistency and inter-tool inconsistency tin this context poses interesting research challenges.

- In this thesis, we focused on studying the quality of sentiment analysis tools and exploiting the results to build a weakly supervised method for sentiment analysis. Furthermore, these results can be extended to other NLP problems such as question answering [131] by considering domain knowledge and semantic similarities between different questions/answers to boost the performance and express the uncertainty.

- Since the inconsistencies are present between both tools and workers as presented in works [88], [130], extending the inconsistency study with a study on the difference between inconsistencies in sentiment analysis tools and crowdsourcing to see how accurate/consistent tools are compared to workers, would be very useful and interesting for several research communities.

- In our work, we used pre-trained word2vec to convert the document to vectors and apply different calculation methods. However, the performances might be improved by applying more precise and new methods for document embedding, such as transformers [132], [133] to generate the document's embedding.

- Domain expert rules are beneficial; hence, enriching those rules by modeling the relations between different domains is an exciting research direction that might improve the model's performance and precision.
- Our work focuses on documents with a single aspect. However, since review documents in most cases contain several aspects, extending the inconsistency resolution mechanism in sentiment analysis to deal with documents with different aspects is the auspicious and challenging work direction.

Bibliography

[1] M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer, "Adversarial example generation with syntactically controlled paraphrase networks", in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, pp. 1875–1885.

[2] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid training data creation with weak supervision", *The VLDB Journal*, vol. 29, no. 2, pp. 709–730, 2020.

[3] H. Ding and E. Riloff, "Weakly supervised induction of affective events by optimizing semantic consistency", in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[4] F. Niu, C. Ré, A. Doan, and J. Shavlik, "Tuffy: Scaling up statistical inference in markov logic networks using an rdbms", *arXiv preprint arXiv:1104.3216*, 2011.

[5] S. Greene and P. Resnik, "More than words: Syntactic packaging and implicit sentiment", in *Proceedings of human language technologies: The 2009 annual conference of the north american chapter of the association for computational linguistics*, Association for Computational Linguistics, 2009, pp. 503–511.

[6] M. Dragoni and G. Petrucci, "A fuzzy-based strategy for multi-domain sentiment analysis", *International Journal of Approximate Reasoning*, vol. 93, pp. 59–73, 2018.

[7] R. Feldman, "Techniques and applications for sentiment analysis", *Communications of the ACM*, vol. 56, no. 4, pp. 82–89, 2013.

[8] C. H. E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text", in *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*, 2014.

[9] E. Kouloumpis, T. Wilson, and J. D. Moore, "Twitter sentiment analysis: The good the bad and the omg!", *Icwsm*, vol. 11, no. 538-541, p. 164, 2011.

[10] Y. Wang, A. Sun, J. Han, Y. Liu, and X. Zhu, "Sentiment analysis by capsules", in *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, International World Wide Web Conferences Steering Committee, 2018, pp. 1165–1174.

[11] M. Tsytsarau and T. Palpanas, "Survey on mining subjective data on the web", *Data Min. Knowl. Discov.*, vol. 24, no. 3, pp. 478–514, 2012.

[12] N. N. Dalvi, A. Machanavajjhala, and B. Pang, "An analysis of structured data on the web", *Proc. VLDB Endow.*, vol. 5, no. 7, pp. 680–691, 2012.

[13] K. Zhao, Y. Liu, Q. Yuan, L. Chen, Z. Chen, and G. Cong, "Towards personalized maps: Mining user preferences from geo-textual data", *Proc. VLDB Endow.*, vol. 9, no. 13, pp. 1545–1548, 2016.

[14] L. Zhu, A. Galstyan, J. Cheng, and K. Lerman, "Tripartite graph clustering for dynamic sentiment analysis on social media", in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014, pp. 1531–1542.

[15] A.-M. Popescu and M. Pennacchiotti, "Detecting controversial events from twitter", in *Proceedings of the 19th ACM international conference on Information and knowledge management*, 2010, pp. 1873–1876.

[16] X. Wang, F. Wei, X. Liu, M. Zhou, and M. Zhang, "Topic sentiment analysis in twitter: A graph-based hashtag sentiment classification approach", in *Proceedings of the 20th ACM international conference on Information and knowledge management*, 2011, pp. 1031–1040.

[17] A. Marcus, M. S. Bernstein, O. Badar, D. R. Karger, S. Madden, and R. C. Miller, "Tweets as data: Demonstration of tweeql and twitinfo", in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, 2011, pp. 1259–1262.

[18] M. Balduini, E. D. Valle, D. Dell'Aglio, M. Tsytsarau, T. Palpanas, and C. Confalonieri, "Social listening of city scale events using the streaming linked data framework", in *ISWC*, 2013.

[19] M. Tsytsarau, S. Amer-Yahia, and T. Palpanas, "Efficient sentiment correlation for large-scale demographics", in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, 2013, pp. 253–264.

[20] M. Tsytsarau, T. Palpanas, and M. Castellanos, "Dynamics of news events and social media reaction", in *KDD*, 2014.

[21] E. C. Dragut, H. Wang, P. Sistla, C. Yu, and W. Meng, "Polarity consistency checking for domain independent sentiment dictionaries", *IEEE Transactions on knowledge and data engineering*, vol. 27, no. 3, pp. 838–851, 2015.

[22] K. Schouten and F. Frasincar, "Survey on aspect-level sentiment analysis", *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 3, pp. 813–830, 2015.

[23] D. Tang, F. Wei, B. Qin, N. Yang, T. Liu, and M. Zhou, "Sentiment embeddings with applications to sentiment analysis", *IEEE transactions on knowledge and data Engineering*, vol. 28, no. 2, pp. 496–509, 2015.

[24] M. Tsytsarau and T. Palpanas, "Managing diverse sentiments at large scale", *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 11, pp. 3028–3040, 2016.

[25] S. Amer-Yahia, T. Palpanas, M. Tsytsarau, S. Kleisarchaki, A. Douzal, and V. Christophides, "Temporal analytics in social media", in *Encyclopedia of Database Systems, Second Edition*, Springer, 2018.

[26] W. Wang, J. Gao, M. Zhang, S. Wang, G. Chen, T. K. Ng, B. C. Ooi, J. Shao, and M. Reyad, "Rafiki: Machine learning as an analytics service system", *Proc. VLDB Endow.*, vol. 12, no. 2, pp. 128–140, 2018.

[27] H. Rong, V. S. Sheng, T. Ma, Y. Zhou, and M. A. Al-Rodhaan, "A self-play and sentiment-emphasized comment integration framework based on deep q-learning in a crowdsourcing scenario", *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[28] X. Ji, "Social data integration and analytics for health intelligence", in *Proceedings VLDB PhD Workshop*, 2014.

[29] E. Cambria, S. Poria, D. Hazarika, and K. Kwok, "Senticnet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings", in *Proceedings of AAAI*, 2018.

[30] G. Fu, Y. He, J. Song, and C. Wang, "Improving chinese sentence polarity classification via opinion paraphrasing", in *Proceedings of The Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, 2014, pp. 35–42.

[31] S. Vosoughi, P. Vijayaraghavan, and D. Roy, "Tweet2vec: Learning tweet embeddings using character-level cnn-lstm encoder-decoder", in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, ACM, 2016, pp. 1041–1044.

[32] J. W. Wei and K. Zou, "Eda: Easy data augmentation techniques for boosting performance on text classification tasks", *arXiv preprint arXiv:1901.11196*, 2019.

[33] J. Risch and R. Krestel, "Aggression identification using deep learning and data augmentation", in *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, 2018, pp. 150–158.

[34] M. T. Ribeiro, S. Singh, and C. Guestrin, "Semantically equivalent adversarial rules for debugging nlp models", in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 856–865.

[35] T. Mahler, W. Cheung, M. Elsner, D. King, M.-C. de Marneffe, C. Shain, S. Stevens-Guille, and M. White, "Breaking nlp: Using morphosyntax, semantics, pragmatics and world knowledge to fool sentiment analysis systems", in *Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems*, 2017, pp. 33–39.

[36] B. Liang, H. Li, M. Su, P. Bian, X. Li, and W. Shi, "Deep text classification can be fooled", *arXiv preprint arXiv:1704.08006*, 2017.

[37] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods", in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, ACM, 2017, pp. 3–14.

[38] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Srivastava, and K.-W. Chang, "Generating natural language adversarial examples", *arXiv preprint arXiv:1804.07998*, 2018.

[39] K. Wang and X. Wan, "Sentigan: Generating sentimental texts via mixture adversarial networks", in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, 2018, pp. 4446–4452.

[40] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank", in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[41] A. Esuli and F. Sebastiani, "Sentiwordnet: A publicly available lexical resource for opinion mining.", in *LREC*, Citeseer, vol. 6, 2006, pp. 417–422.

[42] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification", *arXiv preprint arXiv:1605.07725*, 2016.

[43] E. C. Dragut, H. Wang, P. Sistla, C. Yu, and W. Meng, "Polarity consistency checking for domain independent sentiment dictionaries", *IEEE Transactions on knowledge and data engineering*, vol. 27, no. 3, pp. 838–851, 2014.

[44] Y. Kim, "Convolutional neural networks for sentence classification", *arXiv preprint arXiv:1408.5882*, 2014.

[45] T. Chen, R. Xu, Y. He, and X. Wang, "Improving sentiment analysis via sentence type classification using bilstm-crf and cnn", *Expert Systems with Applications*, vol. 72, pp. 221–230, 2017.

[46] F. Kokkinos and A. Potamianos, "Structural attention neural networks for improved sentiment analysis", *arXiv preprint arXiv:1701.01811*, 2017.

[47] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daumé III, "Deep unordered composition rivals syntactic methods for text classification", 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)*, vol. 1, 2015, pp. 1681–1691.

[48] G. Demartini and S. Siersdorfer, "Dear search engine: What's your opinion about...?: Sentiment analysis for semantic enrichment of web search results", in *Proceedings of the 3rd International Semantic Search Workshop*, ACM, 2010, p. 4.

[49] B. Yang and C. Cardie, "Context-aware learning for sentence-level sentiment analysis with posterior regularization", in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2014, pp. 325–335.

[50] X. Feng, Y. Zeng, and Y. Xu, "Recommendation algorithm for federated user reviews and item reviews", in *Proceedings of the 2018 International Conference on Artificial Intelligence and Virtual Reality*, ACM, 2018, pp. 97–103.

[51] S. H. Bach, M. Broecheler, B. Huang, and L. Getoor, "Hinge-loss markov random fields and probabilistic soft logic", *J. Mach. Learn. Res.*, vol. 18, 109:1–109:67, 2017.

[52] S. Baccianella, A. Esuli, and F. Sebastiani, "Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining.", in *LREC*, vol. 10, 2010, pp. 2200–2204.

[53] J. W. Pennebaker, M. E. Francis, and R. J. Booth, "Linguistic inquiry and word count: Liwc 2001", *Mahway: Lawrence Erlbaum Associates*, vol. 71, no. 2001, p. 2001, 2001.

[54] T. Mullen and N. Collier, "Sentiment analysis using support vector machines with diverse information sources", in *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004.

[55] S. Tan, X. Cheng, Y. Wang, and H. Xu, "Adapting naive bayes to domain adaptation for sentiment analysis", in *European Conference on Information Retrieval*, Springer, 2009, pp. 337–349.

[56] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts", in *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, Association for Computational Linguistics, 2004, p. 271.

[57] J. C. Martineau and T. Finin, "Delta tfidf: An improved feature space for sentiment analysis", in *Third international AAAI conference on weblogs and social media*, 2009.

[58] E. Cambria and A. Hussain, *Sentic computing: Techniques, tools, and applications*. Springer Science & Business Media, 2012, vol. 2.

[59] W. Y. Kim, J. S. Ryu, K. I. Kim, and U. M. Kim, "A method for opinion mining of product reviews using association rules", in *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, ACM, 2009, pp. 270–274.

[60] C. Dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts", in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 69–78.

[61] F. Xing, L. Malandri, Y. Zhang, and E. Cambria, "Financial sentiment analysis: An investigation into common mistakes and silver bullets", in *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 978–987.

[62] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis", in *Mining text data*, Springer, 2012, pp. 415–463.

[63] E. Cambria, D. Olsher, and D. Rajagopal, "Senticnet 3: A common and commonsense knowledge base for cognition-driven sentiment analysis", in *Twenty-eighth AAAI conference on artificial intelligence*, 2014.

[64] E. Cambria, C. Havasi, and A. Hussain, "Senticnet 2: A semantic and affective resource for opinion mining and sentiment analysis.", in *FLAIRS conference*, 2012, pp. 202–207.

[65] L. Deng, J. Wiebe, and Y. Choi, "Joint inference and disambiguation of implicit sentiments via implicature constraints", in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 79–88.

[66] Y. Choi and J. Wiebe, "+/-effectwordnet: Sense-level lexicon acquisition for opinion inference", in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1181–1191.

[67] A. Severyn and A. Moschitti, "Twitter sentiment analysis with deep convolutional neural networks", in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, 2015, pp. 959–962.

[68] L. Luo, X. Ao, F. Pan, J. Wang, T. Zhao, N. Yu, and Q. He, "Beyond polarity: Interpretable financial sentiment analysis with hierarchical query-driven attention.", in *IJCAI*, 2018, pp. 4244–4250.

[69] S. Poria, E. Cambria, and A. Gelbukh, "Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment analysis", in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 2539–2544.

[70] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality", in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[71] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification", in *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[72] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation", in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[73] M. Bautin, L. Vijayarenu, and S. Skiena, "International sentiment analysis for news and blogs.", in *ICWSM*, 2008.

[74] B. Chen and X. Zhu, "Bilingual sentiment consistency for statistical machine translation", in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, 2014, pp. 607–615.

[75] T. G. Dietterich, "Ensemble methods in machine learning", in *International workshop on multiple classifier systems*, Springer, 2000, pp. 1–15.

[76] J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky, "Bayesian model averaging: A tutorial (with comments by m. clyde, david draper and ei george, and a rejoinder by the authors", *Statistical science*, vol. 14, no. 4, pp. 382–417, 1999.

[77] L. Breiman, "Bagging predictors", *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[78] R. E. Schapire, "A brief introduction to boosting", in *Ijcai*, vol. 99, 1999, pp. 1401–1406.

[79] P. G. Ipeirotis, F. Provost, and J. Wang, "Quality management on amazon mechanical turk", in *Proceedings of the ACM SIGKDD workshop on human computation*, 2010, pp. 64–67.

[80] A. Adepetu, A. A. Khaja, Y. Al Abd, A. Al Zaabi, and D. Svetinovic, "Crowdrequire: A requirements engineering crowdsourcing platform", in *2012 AAAI Spring Symposium Series*, 2012.

[81] C. Van Pelt and A. Sorokin, "Designing a scalable crowdsourcing platform", in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, 2012, pp. 765–766.

[82] J. Zhang, V. S. Sheng, J. Wu, and X. Wu, "Multi-class ground truth inference in crowdsourcing with clustering", *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 4, pp. 1080–1085, 2015.

[83] X. Wang, R. Jia, X. Tian, X. Gan, L. Fu, and X. Wang, "Location-aware crowdsensing: Dynamic task assignment and truth inference", *IEEE Transactions on Mobile Computing*, vol. 19, no. 2, pp. 362–375, 2018.

[84] J. Huang, W. Hu, Z. Bao, and Y. Qu, "Crowdsourced collective entity resolution with relational match propagation", in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, IEEE, 2020, pp. 37–48.

[85] Y. Li, B. IP Rubinstein, and T. Cohn, "Truth inference at scale: A bayesian model for adjudicating highly redundant crowd annotations", in *The World Wide Web Conference*, 2019, pp. 1028–1038.

[86] S. Faridani and G. Buscher, "Labelboost: An ensemble model for ground truth inference using boosted trees", in *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, vol. 1, 2013.

[87] J. Zhang and X. Wu, "Multi-label truth inference for crowdsourcing using mixture models", *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[88] Y. Zheng, G. Li, Y. Li, C. Shan, and R. Cheng, "Truth inference in crowdsourcing: Is the problem solved?", *Proceedings of the VLDB Endowment*, vol. 10, no. 5, pp. 541–552, 2017.

[89] Q. Li, Y. Li, J. Gao, L. Su, B. Zhao, M. Demirbas, W. Fan, and J. Han, "A confidence-aware approach for truth discovery on long-tail data", *Proceedings of the VLDB Endowment*, vol. 8, no. 4, pp. 425–436, 2014.

[90] J. Fan, G. Li, B. C. Ooi, K.-l. Tan, and J. Feng, "Icrowd: An adaptive crowdsourcing framework", in *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, 2015, pp. 1015–1030.

[91] D. Zhou, J. C. Platt, S. Basu, and Y. Mao, "Learning from the wisdom of crowds by minimax entropy", in *Proceedings of the 25th International Conference on Neural Information Processing Systems-Volume 2*, 2012, pp. 2195–2203.

[92] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the em algorithm", *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, vol. 28, no. 1, pp. 20–28, 1979.

[93] G. Demartini, D. E. Difallah, and P. Cudré-Mauroux, "Zencrowd: Leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking", in *Proceedings of the 21st international conference on World Wide Web*, 2012, pp. 469–478.

[94] X. Liu, M. Lu, B. C. Ooi, Y. Shen, S. Wu, and M. Zhang, "Cdas: A crowdsourcing data analytics system", *arXiv preprint arXiv:1207.0143*, 2012.

[95] Z.-Y. Zhang, P. Zhao, Y. Jiang, and Z.-H. Zhou, "Learning from incomplete and inaccurate supervision", *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[96] M. Zhou, H. Wang, and K. C.-C. Change, "Learning to rank from distant supervision: Exploiting noisy redundancy for relational entity search", in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, IEEE, 2013, pp. 829–840.

[97] Z. Guan, L. Chen, W. Zhao, Y. Zheng, S. Tan, and D. Cai, "Weakly-supervised deep learning for customer review sentiment classification.", in *IJCAI*, 2016, pp. 3719–3725.

[98] Y. Meng, J. Shen, C. Zhang, and J. Han, "Weakly-supervised neural text classification", in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 983–992.

[99] F. Wu, J. Zhang, Z. Yuan, S. Wu, Y. Huang, and J. Yan, "Sentence-level sentiment classification with weak supervision", in *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, 2017, pp. 973–976.

[100] Y. Wang, W. Yang, F. Ma, J. Xu, B. Zhong, Q. Deng, and J. Gao, "Weak supervision for fake news detection via reinforcement learning", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 516–523.

[101] D. Koller, N. Friedman, S. Džeroski, C. Sutton, A. McCallum, A. Pfeffer, P. Abbeel, M.-F. Wong, C. Meek, J. Neville, *et al.*, *Introduction to statistical relational learning*. MIT press, 2007.

[102] L. Getoor, N. Friedman, D. Koller, A. Pfeffer, and B. Taskar, "Probabilistic relational models", *Introduction to statistical relational learning*, vol. 8, 2007.

[103] K. Kersting and L. De Raedt, "1 bayesian logic programming: Theory and tool", *Statistical Relational Learning*, p. 291, 2007.

[104] P. M. Domingos and D. Lowd, "Unifying logical and statistical AI with markov logic", *Commun. ACM*, vol. 62, no. 7, pp. 74–83, 2019.

[105] D. Koller and N. Friedman, "Structured probabilistic models: Principles and techniques", *MIT Press. To appear*, vol. 48, pp. 54–61, 2009.

[106] D. Lowd and P. Domingos, "Efficient weight learning for markov logic networks", in *European conference on principles of data mining and knowledge discovery*, Springer, 2007, pp. 200–211.

[107] M. Richardson and P. Domingos, "Markov logic networks", *Machine learning*, vol. 62, no. 1-2, pp. 107–136, 2006.

[108] K. Cortis, A. Freitas, T. Daudert, M. Huerlimann, M. Zarrouk, S. Handschuh, and B. Davis, "Semeval-2017 task 5: Fine-grained sentiment analysis on financial microblogs and news", in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 519–535.

[109] R. He and J. McAuley, "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*, International World Wide Web Conferences Steering Committee, 2016, pp. 507–517.

[110] W. B. Dolan and C. Brockett, "Automatically constructing a corpus of sentential paraphrases", in *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.

[111] S. Rosenthal, N. Farra, and P. Nakov, "Semeval-2017 task 4: Sentiment analysis in twitter", in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 502–518.

[112] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, "Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation", *arXiv preprint arXiv:1708.00055*, 2017.

[113] C. Quirk, C. Brockett, and B. Dolan, "Monolingual machine translation for paraphrase generation", 2004.

[114] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances", in *International conference on machine learning*, 2015, pp. 957–966.

[115] K. Denecke, "Using sentiwordnet for multilingual sentiment analysis", in *2008 IEEE 24th International Conference on Data Engineering Workshop*, IEEE, 2008, pp. 507–512.

[116] J. Wiebe, T. Wilson, and C. Cardie, "Annotating expressions of opinions and emotions in language", *Language resources and evaluation*, vol. 39, no. 2-3, pp. 165–210, 2005.

[117] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification", in *Proceedings of the 45th annual meeting of the association of computational linguistics*, 2007, pp. 440–447.

[118] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis", in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, Association for Computational Linguistics, 2011, pp. 142–150.

[119] J. McAuley and J. Leskovec, "Hidden factors and hidden topics: Understanding rating dimensions with review text", in *Proceedings of the 7th ACM conference on Recommender systems*, ACM, 2013, pp. 165–172.

[120] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision", *CS224N project report, Stanford*, vol. 1, no. 12, p. 2009, 2009.

[121] L. E. Bertossi, "Inconsistent databases", in *Encyclopedia of Database Systems, Second Edition*, Springer, 2018.

[122] X. L. Dong and D. Srivastava, "Entity resolution", in *Encyclopedia of Database Systems, Second Edition*, Springer, 2018.

[123] N. Prokoshyna, J. Szlichta, F. Chiang, R. J. Miller, and D. Srivastava, "Combining quantitative and logical data cleaning", *Proc. VLDB Endow.*, vol. 9, no. 4, pp. 300–311, 2015.

[124] E. Krivosheev, S. Bykau, F. Casati, and S. Prabhakar, "Detecting and preventing confused labels in crowdsourced data", *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 2522–2535, 2020.

[125] A. Drutsa, V. Fedorova, D. Ustalov, O. Megorskaya, E. Zerminova, and D. Baidakova, "Crowdsourcing practice for efficient data labeling: Aggregation, incremental relabeling, and pricing", in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2020, pp. 2623–2627.

[126] B. I. Aydin, Y. S. Yilmaz, Y. Li, Q. Li, J. Gao, and M. Demirbas, "Crowdsourcing for multiple-choice question answering", in *Twenty-Sixth IAAI Conference*, 2014.

[127] S. Srinivasan, G. Farnadi, and L. Getoor, "Bowl: Bayesian optimization for weight learning in probabilistic soft logic", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 10 267–10 275.

[128] L. Chou, S. Sarkhel, N. Ruozzi, and V. Gogate, "On parameter tying by quantization", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, 2016.

[129] A. Kalyanpur, B. Parsia, M. Horridge, and E. Sirin, "Finding all justifications of owl dl entailments", in *The Semantic Web*, Springer, 2007, pp. 267–280.

[130] M. J. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin, "Crowddb: Answering queries with crowdsourcing", in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, 2011, pp. 61–72.

[131] S. Shah, A. Mishra, N. Yadati, and P. P. Talukdar, "Kvqa: Knowledge-aware visual question answering", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 8876–8884.

[132] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations", *arXiv preprint arXiv:1909.11942*, 2019.

[133] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", *arXiv preprint arXiv:1810.04805*, 2018.