UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Gustavo Lúcius Fernandes

Challenges in Automatic Peer Review

Belo Horizonte
2022

Gustavo Lúcius Fernandes

Challenges in Automatic Peer Review

Versão Final

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Mestre em Ciência da Computação.

Orientador: Pedro Olmo Stancioli Vaz-de-Melo

Belo Horizonte
2022

Gustavo Lúcius Fernandes

Challenges in Automatic Peer Review

Final Version

Thesis presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Pedro Olmo Stancioli Vaz-de-Melo

Belo Horizonte
2022

Fernandes, Gustavo Lúcius

F363c Challenges in automatic peer review [manuscrito] / Gustavo Lúcius Fernandes..— 2022.
77 f. il.

Orientador: Pedro Olmo Stancioli Vaz de Melo.
Dissertação (mestrado) - Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação
Referências: f. 69-75.

1. Computação – Teses. 2. Revisão por pares – Teses. 3. Classificação de texto – Teses. 4. Aprendizado de Máquina – Teses. 5. Processamento de Linguagem Natural – Teses. I. Melo, Pedro Olmo Stancioli Vaz de. II. Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação. IV.Título.

CDU 519.6*82(043)

Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa
CRB 6/1510 – Insituto de Ciências Exatas da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

CHALLENGES IN AUTOMATIC PEER REVIEW

GUSTAVO LÚCIUS FERNANDES

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores(a):

Prof. Pedro Olmo Stancioli Vaz de Melo - Orientador
Departamento de Ciência da Computação - UFMG

Profa. Helena de Medeiros Caseli
Departamento de Computação - UFSCar

Prof. Marcos André Gonçalves
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 22 de abril de 2022.

Documento assinado eletronicamente por **Pedro Olmo Stancioli Vaz de Melo**, **Professor do Magistério Superior**, em 25/04/2022, às 13:47, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Helena de Medeiros Caseli**, **Usuário Externo**, em 26/04/2022, às 14:22, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Marcos Andre Goncalves**, **Membro de comissão**, em 03/05/2022, às 11:20, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1402946** e o código CRC **8AB22E7C**.

Aos que vieram antes de mim, que lutaram e resistiram para que eu pudesse estar onde estou, e ser quem eu sou.

Acknowledgments

A construção de quem sou hoje é fruto do meu relacionamento com diversas pessoas, dentre elas: familiares, professores, amigos e colegas. Sou grato a cada uma delas por sua contribuição.

No entanto, é justo e necessário destacar o apoio que recebi de alguma delas durante a minha jornada pelo mestrado. Dessa forma, eu agradeço profundamente:

A **Deus**, embora não seja uma pessoa, pela minha vida e por me sustentar durante esses anos;

À minha mãe, **Vânia**, por seu apoio incondicional às minhas escolhas, e por ter cuidado de mim com tanto carinho, amor e dedicação;

Ao meu pai, **Ronaldo**, por estimular a minha curiosidade e incentivar a minha busca pelo conhecimento desde a infância por meio das revistas Recreio, o que foi fundamental para toda a minha vida acadêmica;

À minha companheira, **Aline**, pelos incentivos, por escutar meus desabafos, por me fazer sorrir, por me acolher, por me amar;

Aos meus pais e a Aline, também agradeço por acreditarem na minha capacidade, mesmo nos momentos em que eu duvidava;

Ao meu orientador, **Pedro Olmo**, por todo apoio, ensinamentos e contribuições para a minha formação e para este trabalho;

Aos **amigos da Mocidade Miramez**, pelas preces, pelas conversas e pelas palavras de apoio;

Ao **Bruno Guilherme** e ao **Fabrício Souza**, pelas muitas idas ao laboratório, durante a pandemia de COVID-19, para ligar as máquinas que eu estava utilizando, por acesso remoto, para treinar e avaliar os algoritmos utilizados neste trabalho;

A **cada um dos Gustavos que fui** nesses 25 anos de vida, por suas escolhas, algumas certas, algumas erradas, mas todas me deram experiências para vencer os desafios que encontrei nesses últimos anos.

"*O cérebro eletrônico faz tudo*
Faz quase tudo
Quase tudo
Mas ele é mudo
O cérebro eletrônico comanda
Manda e desmanda
Ele é quem manda
Mas ele não anda..."
(Cérebro Eletrônico - Gilberto Gil (1969))

Resumo

O processo de revisão por pares é o principal recurso acadêmico para garantir que a ciência avance e seja divulgada. Para contribuir com esse importante processo, trabalhos foram realizados para criar modelos de classificação capazes de prever a nota e a decisão final de um artigo a partir do texto do relatório de revisão. No entanto, as tarefas de dar nota e decidir sobre a aceitação ou rejeição de um artigo apresentam diversos desafios tanto para humanos quanto para máquinas. Neste trabalho, nós analisamos o desempenho de modelos estado da arte nestas tarefas quando expostos a instâncias difíceis relacionadas à mudança de texto e nota durante a fase de *rebuttal*, bem como instâncias difíceis relacionadas a revisões *borderlines*. Além disso, discutimos o quão longe estamos de ter um sistema capaz de dar a nota para um artigo e decidir a situação final dele de forma automática. Nossos experimentos mostraram, por exemplo, que o desempenho de um modelo para prever a decisão final de um artigo é 23,31% menor quando exposto a instâncias difíceis e que os classificadores quando erram, cometem esse erro com uma confiança muito alta. Esses e outros resultados nos levaram a concluir que ainda estamos longe de sistemas automáticos para dar notas a artigos e prever a situação final deles por meio do texto dos relatórios dos revisores, no entanto mostramos que as dificuldades enfrentadas pelas máquinas também são enfrentadas por humanos. Isso indica que para a implantação de um sistema de revisão por pares automático, talvez seja necessário repensar o processo de escrita das revisões, para que as impressões e posicionamentos dos revisores sejam mais claras.

Palavras-chave: Revisão por Pares. Instâncias Difíceis. Classificação de Texto. Classificação de Polaridade. Aprendizado de Máquina. Processamento de Linguagem Natural.

Abstract

The peer review process is the main academic resource to ensure that science advances and is disseminated. To contribute to this important process, works were developed to create classification models capable of predicting the score and the final decision of a paper based on the text of the review report. However, the tasks of scoring a paper and deciding whether to accept or reject it present several challenges for both humans and machines. In this work, we analyze the performance of state-of-the-art models in these tasks, when exposed to hard instances related to text and score change during the rebuttal phase, as well as when exposed to hard instances related to borderline reviews. In addition, we discuss how far we are from having a system to score a paper and decide its final status automatically. Our experiments showed, for example, that the performance of a model to predict the final decision of a paper is 23.31% lower when it is exposed to hard instances. We also found that the classifiers make mistakes with a very high confidence. These and other results led us to conclude that we are still far from automatic systems for scoring papers and predicting their final status based on the text of reviewers' reports, however we show that the difficulties faced by machines are also faced by humans. This indicates that for the deployment of an automatic peer review system, it may be necessary to rethink the review writing process, so that the reviewers' impressions and positions are clearer.

Keywords: Peer Review. Hard Instances. Text Classification. Polarity Classification. Machine Learning. Natural Language Processing.

List of Figures

List of Tables

List of Symbols

S_*T_1 Refers to the set of reviews that did not have their score changed and did not have their text changed, and reviews that that had their score changed and did not have their text changed.

S_0T_0 Refers to the set of reviews that did not have their score changed and did not have their text changed.

S_0T_1 Refers to the set of reviews that did not have their score changed and had their text changed.

S_1T_0 Refers to the set of reviews that had their score changed and did not have their text changed.

S_1T_1 Refers to the set of reviews that had their score changed and had their text changed.

S_{++} Refers to the set of reviews associated with *acceptance* scores, that is, scores between 7 and 10.

S_{--} Refers to the set of reviews associated with *rejection* scores, that is, scores between 1 and 4.

S_{0*} Refers to the set of reviews associated with the score 5 or associated with the score 6.

S_{0+} Refers to the set of reviews associated only with the score 6, which denotes an *acceptance borderline* score.

S_{0-} Refers to the set of reviews associated only with the score 5, which denotes a *rejection borderline* score.

$\mathcal{S}_*\mathcal{T}_1$ Refers to the set of papers that **at least one** of its reviews are from the set S_0T_1 or S_1T_1 and **none** are from the set S_1T_0.

$\mathcal{S}_0\mathcal{T}_0$ Refers to the set of papers that **all** its reviews are from the set S_0T_0.

$\mathcal{S}_1\mathcal{T}_0$ Refers to the set of papers that **at least one** of its reviews are from the set S_1T_0.

Contents

Chapter 1

Introduction

1.1 The Peer Review Process

The peer review system is a fundamental academic process that helps to advance the state of the art in all fields of science, documenting and communicating scientific discoveries. Kelly et al. [2014] and Publons [2018] made a description of the basic peer review workflows in their work.

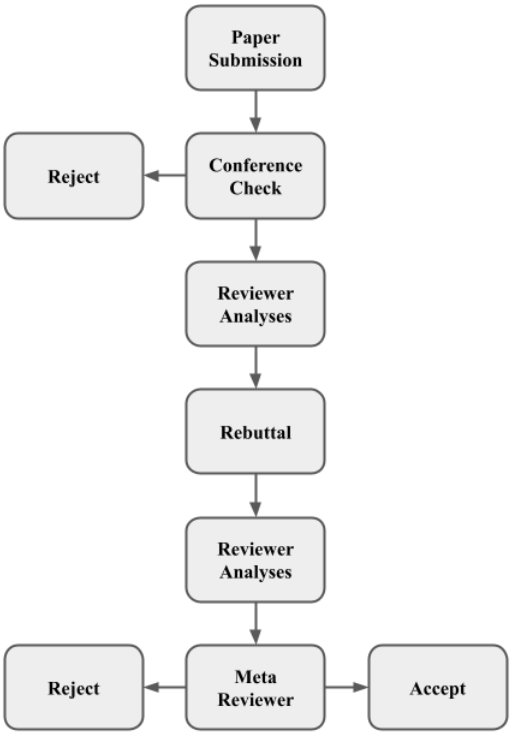
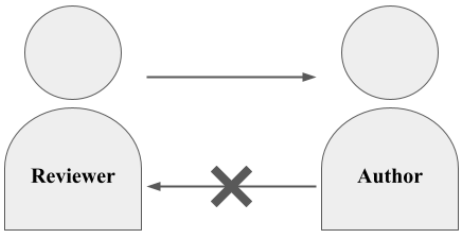


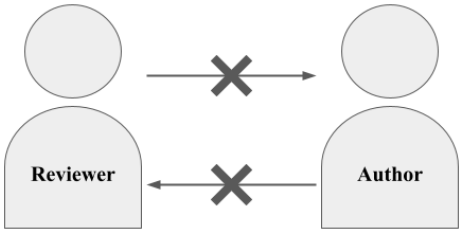
Figure 1.1: Basic peer review workflow.

First, the work is submitted to a conference appropriate to the subject of the paper. Afterwards, the people responsible for the conference review the work to verify if the work addresses issues relevant to the conference, and if it is in accordance with the rules. If the paper fits with what is desired by the conference, the paper is sent to be

peer-reviewed by researchers in the field. If it does not fit, the paper is rejected. After reading the paper, reviewers write a report with their thoughts on the work and give an opinion about the acceptance or rejection of the paper. In the review report, reviewers can provide suggestions and ask questions. In the next phase, called *rebuttal*, reports are sent to authors who have time to adjust what was requested and answer reviewers' questions. This phase is important, because based on what is modified in the work or answered by the authors, reviewers can change the recommendation of acceptance or rejection. After this phase, reviewers give the final verdict on the work. Recommendations are sent to those responsible for the publication, editor or meta-reviewer, who decide whether the paper will be accepted and published or rejected (Figure 1.1).



(a) Single blind interaction.



(b) Double blind interaction.

Figure 1.2: Interactions between reviwers and authors.

There are several types of review processes. According to the Publons report, the two most common types are: *single-blind* review and *double-blind review*. In the single-blind format, the authors do not know who the reviewers are, however the reviewers do have information about the authors (Figure 1.2(a)). In double-blind format authors and reviewers are unaware of each other (Figure 1.2(b)). In this work, we use reviews written during ICLR 2019 and ICLR 2021, years in which the conference used the double-blind format.

1.2 Papers and Reviewers

Despite its importance, this process faces several challenges, such as finding qualified reviewers to meet the growing demand for reviews. A study developed by White [2019], analyzing journals and conferences proceedings in Science and Engineering and indexed in Scopus, shows that from 2008 to 2018 there was an average increase of 3.8 in research published in the world. In 2018, the scientific global production grew from 1.8 million, in 2008, to 2.6 million papers.

Kovanis et al. [2016] developed the first work whose objective is to measure the sustainability of the peer review system. A mathematical model, fed with data extracted from a bibliographic database called MADLINE, was used to estimate the quality of the review process in terms of the number of reviewers available. The authors concluded that the availability of reviewers outweighs the demand for reviews in all tested scenarios. Meanwhile, the Publons [2018] report shows that for a publisher to get a review on a paper, it is necessary to make an average of 2.4 invitations and, in 2025, this number is expected to rise to 3.6 invitations. These two studies suggest that there is a high number of possible reviewers available, however, the tendency is that the refusal to this task increases, making the review process slower. According to Publons, the lack of time is one of the biggest reasons for declining review invitations. This is understandable, since the average time taken to carry out a review is 19.1 days [Publons, 2018]. During this time, reviewers have to read the paper, write a report on the work, and give it a score. The latter, although not the only factor, has a very important role in the final decision regarding the paper acceptance. Chakraborty et al. [2020] showed that the score has a correlation of about 90% with the final decision to reject or accept the paper.

1.3 Motivation

While it sounds simple, giving a fair score during the review process is a complex task. On one hand, because the reviewer's perception of the paper's scientific strengths and weaknesses only depends on their own experience and research background, the text of the review *should not change much* with the venue of the publication. On the other hand, because conferences and journals have different levels of acceptance rates and demands, the score given by the review might vary accordingly. This is why it is common for a paper to be rejected at one conference and accepted at another without major changes

in the work [Smith, 2006]. For instance, a review pointing to a weakness related to the lack of statistical significance tests in a given empirical result might be sufficient for a rejection score in a highly competitive conference but not critical enough in other less competitive conferences. The fact is that with the different reviewers' backgrounds and different conferences' levels of acceptance rates and demand, even after the text of the review is finished, the reviewer may be in doubt about the score that does more justice to the text of the review and to be in doubt between accepting or rejecting the paper, which makes the reviewer task even more difficult. In some cases, the paper is clearly positioned on a thin line that separates it from rejection and acceptance, and which side the paper goes on may depend on a score decision made with very little confidence. Papers around this thin line between approval and rejection are usually called *borderline papers*, and for many papers of this type, review scores and texts may be inconsistent [Gao et al., 2019].

In order to contribute to this process and help the reviewers and meta-reviewers, classification models were created for two tasks: **(1) predict the score of a paper** and **(2) predict the acceptance decision of a paper** (both highly dynamic values) **from the texts written by the reviewers** (usually static information) [Ghosal et al., 2019a; Deng et al., 2020]. However, just like for humans, this classification task is also non-trivial and has several challenges involved. As we will show in this work, at different stages of the review process, the same text can be associated with different scores, and this can be a problem for supervised learning models. This change occurs in cases where, after the reviewer releases the text of the review with its score, the authors have a period, called *rebuttal*, to respond to the text written by the reviewers, clarify doubts and defend the work. After this phase, the reviewer can change the score according to the author's answer. We will treat the period before rebuttal as *pre-rebuttal* and after *post-rebuttal*. Another challenge is to accurately identify text reviews recommending borderline scores (e.g. weak accept). Because these reviews make recommendations that are close to both acceptance and rejection, they tend to have similar sentiments and features even when they are recommending opposing scores, i.e., weak accept and weak reject. In other words, this can make it difficult for models to learn meaningful features to differentiate the positive borderlines from the negative borderlines. Unfortunately, it is common for reviewers to have to deal with this situation, as the highest concentration of scores in the most popular conferences are usually located in borderline scores [Deng et al., 2020].

Thus, before deploying an automatic peer review system, it is crucial to carefully evaluate these classification models, especially with regard to the data used in the learning process and their confidence when making a prediction. Therefore, it is essential to study these difficult cases to understand the impact they have on existing classifiers. That way, we will know if the performance of the models is being overestimated or underestimated because of instances of certain types. Furthermore, with this impact mapped, models will be developed so that their results are fairer and more transparent, which can help

reviewers mitigate their errors and not propagate them. Unfortunately, until recently, the lack of public databases containing review scores and texts made it difficult for this type of work to be carried out. Currently, some databases containing the text, score and final decision of a reviewer are already available, however, only one [Gao et al., 2019] has *pre-rebuttal* and *post-rebuttal* records. Nevertheless, it is still unknown how much the reviews that underwent these changes can influence the performance of classification models.

1.4 Goals

The peer review process is very important, however its success is directly related to the performance of the reviewers, since they are the ones who score the papers and decide which ones will be accepted or rejected. To make the process more efficient and fair, works were developed to create models that predict the paper's score and its final status (accepted or rejected), based on the review report written by the reviewers and, from that, can guide the reviewers' decisions. Therefore, it is important to investigate which scenarios these models may have more problems with. Based on this, the goals of this work are:

1. Evaluating the performance of the state-of-the-art classification models in the tasks of (1) predicting the score of a paper and (2) predicting the final acceptance decision of a paper, when exposed to *hard instances*. Our hypothesis is that *hard instances* negatively impact the performance of classifiers.

2. Discussing how close we are from systems in which the reviewers only write their reviews and, from that, the score and the decision about the acceptance or rejection of the paper is made automatically.

The models assessed in our work (*DeepSentiPeer* [Ghosal et al., 2019a], *HabNet* [Deng et al., 2020], *C-LSTM* [Zhou et al., 2015], *CNN-GRU* [Wang et al., 2016], *BERT* [Devlin et al., 2019], *RoBERTa* [Liu et al., 2019] and *XLNet* [Yang et al., 2019]) classify the review based on the complete text, and do not perform an aspect-based classification [Brauwers and Frasincar, 2021]. That means that the models do not identify the aspect (i.e. clarity, originality, results) that the reviewers are evaluating.

Our experiments are conducted on data collected from the *OpenReview.net* website. We collected reviews submitted to the ICLR conference in the years of 2019 and 2021. We chose these two years because only in them we were able to collect data in the *pre-rebuttal* and *post-rebuttal* phases. In total, we collected 14,459 reviews containing their texts and

scores before and after the rebuttal. Each review is associated with a paper, which in turn is associated with an acceptance or rejection decision. We only use reviews of papers that have a decision and we only use papers that have all their reviews, totaling 4,035 papers.

1.5 Contributions

The tasks developed to achieve the objectives proposed in this work resulted in the following contributions:

1. We present the general aspects of the data set we have collected and describe its data distribution when taking into account the different states that a review can have over time (through the *pre-rebuttal* and *post-rebuttal* phases).
2. We assess the overall performance of models created specifically for classifying scientific paper reviews and models created for classifying text in general.
3. We assess the performance of models when exposed to instances considered more difficult due to changes in the text and in the score, during the *pre-rebuttal* and *post-rebuttal* phases, to measure the impact they have on the classification process.
4. We assess the performance of models when exposed to borderline reviews, and the impact of changing borderline acceptance and rejection scores (e.g. "weak accept") to pure borderline scores (e.g. "neutral") in the final decision of a paper.
5. We evaluate the errors made by the classifiers in order to understand how far these models are from automatically executing the *review score prediction* and *paper decision prediction* tasks without the need for human intervention.

1.6 Dissertation Organization

This dissertation is organized as follows:

- **Chapter 2** provides some background definitions about text representation, text classification and peer review process.

- **Chapter 3** provides a review of related work that address solutions developed to help the peer review process, text classifiers used in peer review process and in others domains, and hard instances.
- **Chapter 4** describes the *review score prediction* and *paper decision prediction* problem. Besides that, it also describes the types of hard instances explored in this work.
- **Chapter 5** provides the experimental setup of the work, describing the data set and the models analyzed.
- **Chapter 6** reports the experiments executed to measure the overall performance of the classifiers and the impact of hard instances on the classification process.
- **Chapter 7** reports the investigation about how close are the models to solving the *review score prediction* and *paper decision prediction*.
- **Chapter 8** concludes this dissertation recapturing the results, answering the hypotheses and presenting future work.

Chapter 2

Background

2.1 Text Representation

Language, written or spoken, is a tool by which much information is produced. According to a global survey carried out by DataReportal [2020], in January 2020 around 3.81 billion people accessed at least one social media, of which three of the five most popular are intended for exchanging messages, they are: WhatsApp (2 Billions users), Facebook Messenger (1.3 Billion users) and WeChat (1.16 Billion users). However, in order for it to be used for machine learning, it is first necessary to represent it in an adequate way to be interpreted by algorithms.

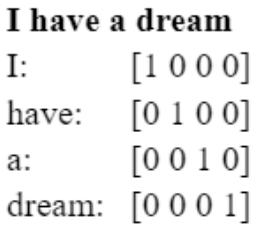


Figure 2.1: Representation of the sentence "I have a dream" in one-hot encoding.

One way to make this representation is using one-hot encoding [Cerda et al., 2018]. In this approach, considering a vocabulary of size N, each word is represented by a vector of size N filled with 1 in the position referring to the word and 0 in the other positions [Kowsari et al., 2019]. The Figure 2.1 shows a sentece[1] and its representation as one-hot enconding. In this scenario, we consider a hypothetical four-word vocabulary.

Another way to represent a text is through a term-frequency vector [Salton and Buckley, 1988]. In this approach, considering a vocabulary of size N, the text is represented by a vector of size N, where each position represents a word. The vector is then filled with the number of times a word appears in the text. Figure 2.2 shows a sentence[2] and

[1] Martin Luther King Jr.' quote
[2] Archbishop Desmond Tutu' quote

its representation as a term-frequency vector. In this scenario, we consider a hypothetical vocabulary compound only by the words of the sentence.

If you are neutral in situations of injustice, you have chosen the side of the oppressor.

if	you	are	neutral	in	situations	of	injustice	have	chosen	the	side	opressor
1	2	1	1	1	1	2	1	1	1	2	1	1

Figure 2.2: Representation of the sentence "If you are neutral in situations of injustice, you have chosen the side of the oppressor." in term-frequency vector.

Using also the frequency of words, it is possible to represent a text with a TF-IDF (**T**erm-**F**requency **I**nverse **D**ocument **F**requency) vector [Ramos et al., 2003], but, in this case, the frequency of words in other texts is also considered. In this approach, first is calculated how often a word w appears in a document d (Equation 2.1). Then, the inverse frequency of the document is calculated, to measure the importance of this term when considering the other texts (Equation 2.2). Lastly, the TF-IDF is the multiplication of the term-frequency by the inverse frequency of the document (Equation 2.3).

$$TF_{w,d} = \frac{\text{quantity of word } w \text{ in } d}{\text{number of words in } d} \tag{2.1}$$

$$IDF_w = \log(\frac{\text{number of documents}}{\text{number of documents with word } w}) \tag{2.2}$$

$$TF - IDF_{w,d} = TF_{w,d} * IDF_w \tag{2.3}$$

The approaches presented so far represent the text with a lack of semantic information. For example, although *small* and *little* have similar meanings, there is nothing in the one-hot vector, term-frequency vector, or TF-IDF vector to show this. Another problem is that the word order in the sentences is not considered, so the sentences *make love, not war* and *make war, not love* will be represented in the same way. However, it is possible to represent texts in a way that semantic information is considered, using word embeddings [Goldberg, 2016]. This technique uses algorithms called Neural Network to learn features of words and transform them into a vector of real number. Many methods have already been used to build word embeddings, some of them are Word2Vec [Mikolov et al., 2013], and FastText [Bojanowski et al., 2017].

2.2 Text Classification

The Text Classification task can be defined as an automatic categorization of a text into one or more predefined classes, based on its content [Joseph and Ramakrishnan, 2015]. This classification is considered a supervised machine learning technique, because a group of texts already categorized is used for machine learning [Kadhim, 2019]. Therefore, a group of texts $T = \{t_1, \ldots, t_n\}$ with pre-defined classes $C = \{c_1, \ldots, c_m\}$ is used to learn how to classify a new document d, into one or more classes in C [Allahyari et al., 2017].

Kowsari et al. [2019] highlight in their work four main levels to which textual classification can be applied:

1. **Document Level:** The algorithm classify the complete document into some category.
2. **Paragraph Level:** The algorithm classify a paragraph (a portion of a document) into some category.
3. **Sentence Level:** The algorithm classify a sentence (a portion of a paragraph) into some category.
4. **Sub-sentence Level:** The algorithm classify an expression (a portion of a sentence) into some category.

Many approaches were used for Text Classification (Naïve Bayes Classifier [Rish et al., 2001], Support Vector Machine [Noble, 2006], K-Nearest Neighbor [Keller et al., 1985]), but the Deep Learning models achieved better results in several Natural Language Processing (NLP) tasks, including Text Classification, simulating the human brain to learn characteristics of data [Kowsari et al., 2019]. The basic structure of a Deep Neural Networks consists of one input layer, two or more hidden layers and one output layer (Figure 2.3). The input layer receives the text representation and transfers it to the hidden layers. In these layer, neurons perform calculations and transfer the results to the output layer, the last layer. The output layer contains values that represent the network's response to a given task [Kowsari et al., 2019]. In the case of Text Classification, the final layer represents the class to which a given text belongs.

The training of a Deep Learning Model, consists of taking the result of the final layer, comparing it with the real class of an instance and checking how far they are from each other. This distance is called *error*. The next step, called *backpropagation*, is to use the error found to adjust the hidden layer weights with the purpose of calculating them to generate results that make the error of the final layer decrease. Training ends

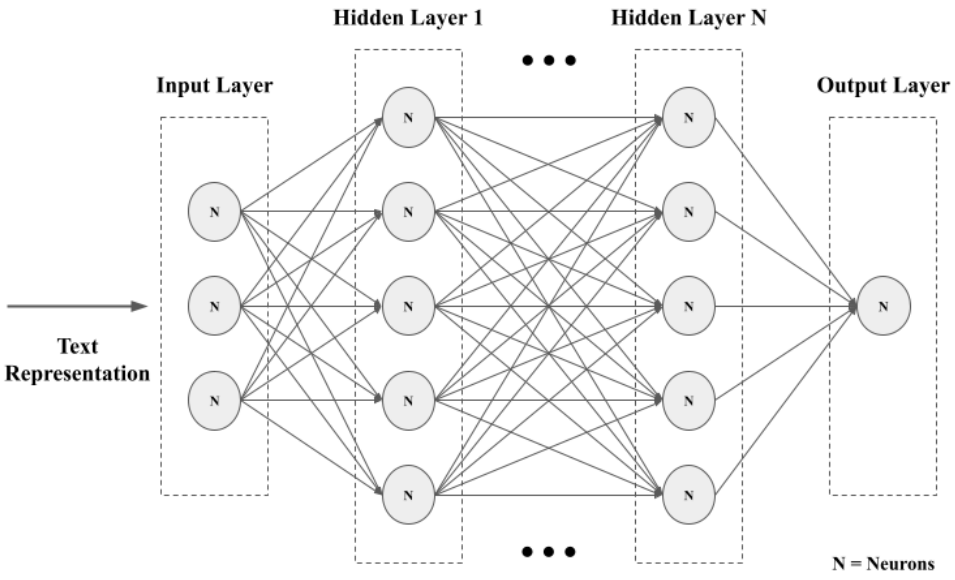


Figure 2.3: An illustration of a basic structure of a Deep Neural Network.

when it reaches a certain condition, for example, training may stop when the learning process repeats a predetermined amount of times, or until the error no longer decreases. Other architectures emerged from this idea (CNN [Goldberg, 2016], LSTM [Hochreiter and Schmidhuber, 1997], BERT [Devlin et al., 2019]).

Chapter 3

Related Work

3.1 Contributions to the Peer Review Process

Due to the importance of the peer review process, several works have already been carried out with the aim of analyzing the process and contributing to make it more transparent, fair and efficient. For example, Ragone et al. [2013] analyzed the quality of the peer review process at computer science conferences and identified the presence of acceptance and rejection biases related to gender, author affiliation and geographic location. It also highlights the presence of reviewers who tend to always give high or low scores, which is known as *rating bias*. In this direction, Tomkins et al. [2017] also analyzed the bias related to author affiliation, showing that in the single blind review process, the fame of the authors or the institutions they belong gives a great advantage for acceptance. In their works Silva and Vance [2017] discussed these biases, presenting the opinions of several authors about them, with the aim of understanding the problems and deficiencies of the review process.

During the NIPS conference, in 2014, Langford and Guzdial [2015] verified the consistency of the process. They distributed 10% of the papers submitted to the conference to be evaluated by two different groups of reviewers and noted that the groups diverged in terms of acceptance or rejection in more than a quarter of the papers. Chakraborty et al. [2020] used sentiment analysis to explore the correlation between the characteristics of a paper and its final decision. The object of study was review's texts from the ICLR conference (2017, 2018 and 2019). They find that the appropriateness and the clarity are the most relevant aspects in both acceptance and rejection.

In order to simplify and improve the assertiveness of an important step in the review process, Charlin and Zemel [2013] and Anjum et al. [2019] develop systems that associate papers with most suitable reviewers to evaluate them. Seeking to help editors exclude papers outside the scope of the conference, Ghosal et al. [2019b] developed a multiview clustering which, trained with previous papers, was able to predict whether a new paper was within the scope of a given conference. The model was tested using

lexical, semantic and bibliographic representations of the papers. Results were generated for each of these representations and combinations between them. The best result was achieved when using the combination between the lexical, semantic and bibliographic representations of the paper, having an accuracy above 90% in the decision to keep or withdraw a paper.

Exploring the content of review texts, Kumar et al. [2021a] developed a classifier to perform aspect extraction and sentiment classification of sentences from a review. Thus, given a sentence, the model is able to predict the aspect to which it refers (e.g. clarity) and the sentiment polarity of the sentence (e.g. positive). In their work, Singh et al. [2021] presented a database called *COMPARE* composed of sentences extracted from reviews, that indicate to the papers' authors the absence or presence of important references in the paper. Singh et al. [2021] also use pre-trained language models and fine-tune them in the *COMPARE* database to measure their performance in the task of identifying sentences of this type in new reviews.

In order to assist in the meta-review phase, Bhatia et al. [2020] and Kumar et al. [2021b] proposed models to automatically generate the meta-review text, a text that contains a summary of the reviewers' opinion and the final decision of the papers, facilitating the work of the meta-reviewer. Seeking to contribute to the process, Bartoli et al. [2016] proposed a deep neural network framework to generate review comments with a positive, neutral or negative tone, given a paper. According to the work, approximately 30% of the reviews generated were considered genuine by human reviewers. Furthermore, Gipp et al. [2017] and Weber and Karcher [2020] proposed systems to increase the transparency and security of the review and research processes.

3.2 Automatic Review Classification

3.2.1 Reviews in Other Scenarios

Researchers tried to use other approaches to improve the review process, and as we do in this work, there are studies that evaluate and propose classification models to decide whether a paper will be accepted at conferences based on their reviews. Classifiers are already widely used in reviews of other scenarios and have achieved good accuracy results.

In their work, Tsutsumi et al. [2007] created a model capable of classifying movie

reviews' opinions into positive and negative sentiment. Sagar et al. [2021], in the market domain, evaluated and compared different approaches to classify product reviews. Al-Smadi et al. [2018] explored the hotel reviews domain, with the aim of identifying in a customer's text which hotel service the text is referring to, and also the sentiment polarity of the comment. Finally, Ganu et al. [2009] classified sentences of restaurant reviews, according to the topic addressed (e.g. food, service, price). They also classified these sentences in positive, negative or neutral sentiment and sentences with mixed polarity.

It is possible to make an analogy between the sentiment expressed in a movie review (negative and positive) and the paper's final status (rejected and accepted), or between the number of stars given to a product and the score given to a paper. The customer task and the reviewer task are similar, that is, to evaluate something. Nonetheless, differently from Al-Smadi et al. [2018] and Ganu et al. [2009] that identify the aspect of the hotel and restaurant that is being evaluated in the review, the classifiers used in our work classify the entire text of the review without identifying the aspects of the paper that the reviewer is referring to.

3.2.2 Challenges

Scientific paper reviews texts have different characteristics from the previously mentioned review texts, which makes the task more difficult. As already mentioned by Deng et al. [2020] and Wang and Wan [2018], scientific review texts have a larger size, since reviewers have to write a more detailed opinion of several aspects of the evaluated paper. Scientific reviews also have a mix of opinion text with non-opinion text, as reviewers write in the report a summary of the paper and excerpts from the work (non-opinion) and highlight strengths and weaknesses of the work (opinion). In adition, scientific reviews usually mix positive and negative points, since it is normal for reviewers to have positive opinions on some aspects, and negative opinions on others on the same paper, even when the score is extremely positive or negative. All of these make it difficult for models to capture the overall (or general) polarity of the review. Thus, efforts were made to develop classifiers specifically designed for this type of text.

Another difficulty in the supervised classification of paper reviews is obtaining labeled data containing review texts with their corresponding scores, and the final papers' situation (accepted or rejected). Therefore, some efforts were made to meet this need and enable the study and proposition of classifiers. Soergel et al. [2013] developed the platform called *OpenReview.net*, which serves as a communication channel between reviewers, meta-reviewers and authors. The data that is stored on the platform is made

available through an API[1]. Kang et al. [2018] made available the first public database containing 10,770 scientific reviews and suggested some tasks that could be explored from it, such as predicting the score and the final decision of a paper. Hua et al. [2019] created a database with 14,200 reviews, of which 5,050 are associated with the reviewers' score, in order to understand the content and structure of the reviews. A different effort was made by Gao et al. [2019], who provided database containing the text and scores before (1,213 reviews) and after (1,275 reviews) the rebuttal phase of the ACL 2018 conference. They also provided the text written by the papers' authors during the rebuttal phase. In their work, they analyzed the importance of the rebuttal phase to change the reviewer's scores and found that the persuasion, politeness and specificity of the authors' response has a statistically significant effect on changing the scores, especially for borderline papers. However, score alignment among reviewers is the most important factor for a change to occur.

3.2.3 Classifiers of Scientific Paper Reviews

From these and other efforts, classifiers were created specifically for the task of predicting the review score and the final decision of the paper. Kang et al. [2018], made the baselines available for *review score prediction* and *paper decision prediction* tasks, using the architectures: Convolutional Neural Networks (CNN) [Zhang et al., 2015], Recurrent Neural Networks (LSTM) [Hochreiter and Schmidhuber, 1997] and Deep Averaging Networks (DAN) [Iyyer et al., 2015]. Then, Ghosal et al. [2019a] improved the results achieved using an architecture that uses information from the paper and sentiments associated with the review, extracted in an unsupervised way.

In another initiative, Deng et al. [2020] proposed a neural architecture that uses three levels of encoders for text representation in addition to an attention mechanism. The data provided by Kang et al. [2018] were used by these two works, but other reviews needed to be collected to run their experiments. Authors of other works already cited here (Bhatia et al. [2020] and Kumar et al. [2021b]), also sought to use the text of the reviews to predict the final decision of the paper. These models were created with the aim of making the review process fairer and more transparent, as they can guide the reviewer's decision, thus removing the subjective factor from the score and also preventing texts and scores from not corresponding to each other.

[1] https://openreview-py.readthedocs.io/en/latest/

3.3 Hard Instances

Another serious problem inherent to text classification tasks is the unreliability of the data, which can present instances that, for some reason, do not transmit their classes clearly. These instances, known as *hard instances* can, therefore, overestimate or underestimate the effectiveness of the classifiers. Smith et al. [2014] presented an idea of what a hard instance would be, and defined them in the classification aspect of the machine, that is, these instances are those that lead more classifiers to error. On the other hand, Beigman Klebanov and Beigman [2009] defined hard instance in the database annotation aspect, that is, when an instance is difficult to be annotated because it causes doubt and confusion among the annotators, it is a hard instance. In another work, Beigman Klebanov and Beigman [2014] showed that these hard instances can hinder even the learning of easy instances.

In the context of scientific paper reviews classification, the two concepts of hard instances can be applied. The definition that explores machine classification is applicable when thinking about existing classifiers and what types of instances make learning more difficult for them. This aspect is important because we are looking for these instances, and we aim to assess their impact.

The definition that explores the database annotation is applicable when we think about the peer review process. The reviewer has as the final artifact of her/his assessment the review text and a score, so the reviewer herself/himself labels the text she/he has written with a score. It is desirable that this score is in line with the text that was written and expresses the reviewer's perception of the paper. However, as we said before, scoring is not an easy task, which can lead the reviewer to have doubts about which score to give to a particular paper, or worse, to give a score that is in disagreement with the text of the review. This point is important for our work, as the input for learning the supervised models is the review text and the score given by the reviewer.

Martins et al. [2021] proposed a methodology for finding hard instances in movie reviews and showed that such instances are significantly more difficult to classify. To the best of our knowledge, only Wang and Wan [2018] evaluated the impact of hard instances on the *review score prediction* task. In their work, the authors proposed a new neural architecture to predict the score of reviews and the final decision of the paper. More important for the purpose of our work, they also showed that borderline reviews (hard instances) are much harder to classify.

Unlike the work of Wang and Wan [2018] and others cited in this section, in this work (i) we evaluated the overall performance of state-of-the-art models in *review score prediction* and *paper decision prediction* tasks. Furthermore, we evaluated the impact of two types of hard instances on these tasks: (ii) reviews that had their scores and/or texts

changed during the rebuttal phase and (iii) borderline reviews. Regarding the latter, we also evaluated the possibility of replacing borderline acceptances and rejections scores (e.g. weak accept) by a single borderline score. Finally, (iv) we investigate how close are state-of-the-art models to solving the task of classifying reviews and the final acceptance decision of papers. Note that our goal is not to propose a new model for review classification, but to investigate how the current state of the art behaves, especially in challenging scenarios and problematic data.

Chapter 4

Problem Setting

4.1 Problem Design

In this work, we seek to analyze the impact of hard instances on peer review classification models and understand how far we are from having a system that helps reviewers to score reviews and editors to decide whether a paper should be accepted or not. To do this, we identify some instances that may be difficult to classify (*hard instances*) and measure their impact on classification models. We will investigate two prediction tasks: *review score prediction* (*RSP*) and *paper decision prediction* (*PDP*).

4.1.1 Review Score Prediction

The *review score prediction* (*RSP*) task concerns predicting a review's score from its text. Let $\mathcal{R} = \{r_1, r_2, \ldots\}$ be a set of paper reviews where each review $r_i = (t_i, s_i)$ is associated with a text t_i and a score s_i. The objective of the *RSP* task is to learn a classifier $\mathcal{F}(\mathcal{R})$ from $\mathcal{R}$ capable of predicting a $\hat{s}_i$ score for the text of review t_i.

In our dataset, reviews have scores from 1 to 10, but we understand that it is not necessary for the classifiers to make such a harsh prediction, so we group together scores that are similar within the context of acceptance and rejection of the paper. The first set, which we will call the set S_{--}, is composed of reviews associated with *rejection* scores, that is, scores between 1 and 4. The second set, which we will call S_{0-}, is composed of reviews associated only with the score 5, which denotes a *rejection borderline* score. The third set, which we will call S_{0+}, is composed of reviews associated only with the score 6, which denotes an *acceptance borderline* score. Finally, the fourth set, which we will call S_{++}, is composed of reviews associated with *acceptance* scores, that is, scores between 7 and 10. This approach allows for a clear division between scores that represent

acceptance and rejection and, between these two sets, there is a division between what is a clear position, whether acceptance or rejection, and a dubious position, in the case of borderlines (Table 4.1).

Rejection		**Acceptance**	
1 - 4	5	6	7 - 10
Clear Position	**Dubious Position**	**Dubious Position**	**Clear Position**

Table 4.1: Division of scores considering acceptance/rejection and clear/dubious position.

4.1.2 Paper Decision Prediction

The *paper decision prediction* (*PDP*) task concerns predicting the decision to accept a paper from its reviews. Let $\mathcal{P} = \{p_1, p_2, \ldots\}$ be a set of papers submitted for peer review. Each paper $p_i = (R_i, d_i)$ is associated with a set of reviews $R_i = \{r_i^1, r_i^2, \ldots\}$ and a final decision $d_i \in \{0, 1\}$ about its acceptance, where 0 denotes that the paper has been rejected and 1 that has been accepted. As in the *RSP* task, each review $r_i^j = (t_i^j, s_i^j) \in R_i$ is associated with a text t_i^j, which describes the j-th opinion about the paper p_i, and a score s_i^j, which is the corresponding score. The objective of the *PDP* task is to learn a classifier $\mathcal{F}(\mathcal{P})$ from $\mathcal{P}$ capable of predicting a decision $\hat{d}_i$ for a paper p_i.

4.2 Hard Instances

The ideal scenario for classification models to perform well in *RSP* and *PDP* tasks is one in which hard instances do not exist in both training and production phases, that is, in their real-world applications. This means that it is desirable that all scores s_i correspond to the actual opinion described in the t_i texts (*RSP*) and that all final decisions d_i correspond to the actual opinion expressed in the R_i review set of paper p_i (*PDP*).

4.2.1 Review States

However, as seen from the set of reviews we collected, a review can have different states over time, which, as we hypothesize, can impact the performance of classifiers in *review score prediction*. During the rebuttal phase, the reviewer has the option to change the score given to the paper and, with that, change the text of the review. In a worse scenario, the reviewer can also change the score and leave the text unchanged. To differentiate reviews with different change states, we define review sets according to the type of change they have gone through. Sets will be denoted by the notation S_iT_j, where S_i refers to the existence or not of a change in the score and T_j refers to the existence or not of a change in the text.

In the first set are the reviews of the state S_0T_0, that is, this set contains reviews that have not changed either in the score (S_0) or in the text (T_0). In this case, it is more likely that the text t_i is correctly associated with the score s_i given by the reviewer. In the second set are the reviews of the state S_1T_1, that is, reviews that suffered alterations in the text and in the corresponding score. Reviews of this set may still be consistent, that is, the score s_i may be representative of the content of the text t_i, but it is possible that the text changes are not sufficient to represent the score change. The third state contains reviews from the set S_1T_0, that is, the reviews have not changes in the text t_i, but have changes in the score s_i. This case is more difficult than the previous one, since the text t_i hardly corresponds faithfully to the score s_i given by the reviewer. The fourth set, S_0T_1, contains reviews that have not changes in score and have changes in text. In this case, the text has probably changed with some additional explanation of the reviewer's assessment, and possibly the text is even more informative about the score than the previous text. We will use the $*$ symbol to group reviews from different sets. For example, the set S_*T_1 refers to $S_1T_1 \cup S_0T_1$, that is, reviews that have changes in text and scores and reviews that have changes in text and not in score. Note that S_*T_1 corresponds to the set containing all the instances we consider difficult (*hard*) according to their states.

Our first hypothesis is that the performance of review score classifiers (*RSP*) is significantly impacted by review states. Our second hypothesis is the presence of reviews from different states also impacts the final decision regarding the paper (*PDP*). As mentioned in Chapter 1, Chakraborty et al. [2020] showed a strong correlation between the scores $\{s_i^1, s_i^2, \ldots\}$ of the reviews of a paper p_i and its final decision d_i. Therefore, we also associate the previously defined sets to characterize papers according to the types of reviews they receive. A paper belongs to the set $\mathcal{S}_0\mathcal{T}_0$ (or *easy instance*) if all its reviews are from the set S_0T_0. A paper belongs to the set $\mathcal{S}_*\mathcal{T}_1$ (or *hard instance*) if **at least one** of its reviews are from the set S_0T_1 or S_1T_1 and none are from the set S_1T_0. Finally, a paper belongs to the set $\mathcal{S}_1\mathcal{T}_0$ (or *hardest instance*) if **at least one** of its reviews are from

the set S_1T_0.

4.2.2 Review Scores

In addition to the change states that reviews can go through, a specific scenario where text and score may not match closely is the case of reviews with *borderline scores*. Under this circumstance, reviewers may have doubts between acceptance and rejection [Gao et al., 2019]. Moreover, a reviewer may write a text with several sentences characteristic of an acceptance but assign a borderline rejection score to the paper (or vice versa). In these cases, it is evident that the text (t_i) is not consistent with the score (s_i). This mismatch between text and score may impact the prediction of the review score and also the final decision regarding the acceptance of the paper.

Thus, we will assess the impact of the review score classifiers (RSP) on the four review sets: S_{--}, S_{0-}, S_{0+} and S_{++}. While reviews of the sets S_{--} and S_{++} can be considered *easy instances*, reviews of the sets S_{0-} and S_{0+} can be considered *hard instances* to be classified. Also, for the *paper decision prediction* task, we will group instances of the sets S_{0-} and S_{0+} into a new set S_{0*} to understand whether treating borderlines as a single set does not harm performance of classifiers. Such design decision can ease the task of reviewers because assigning a single borderline score (e.g. *neutral*) to a paper is a much easier task for the reviewer than having to decide if a *borderline* paper should be accepted (e.g. weak accept) or rejected (e.g. weak rejection) at a conference.

Chapter 5

Experimental Setup

5.1 Dataset

We extracted our dataset from *OpenReview.net*. The data refers to reviews of papers submitted to the ICLR conference in 2019 and 2021. In total, we collected data from 14,459 reviews and 4,035 papers. From 2019 there are 4,358 reviews and 1,419 papers and from 2021 there are 10,101 reviews and 2,616 papers. The main metadata collected were:

- **Review Texts**: The text of the review report. Before the rebuttal phase, reviewers write their impressions about the paper, can ask questions to authors and write suggestions. Based on what the authors write in the rebuttal phase, reviewers can complement the review text.
- **Scores**: The score given to papers by reviewers. The ICLR conference gives the following meaning to each score: (1) Trivial or wrong; (2) Strong reject; (3) Clear rejection; (4) Ok, but not good enough - Rejection; (5) Marginally below acceptance threshold; (6) Marginally above acceptance; (7) Good paper; (8) Top 50% of accepted papers; (9) Top 15% of accepted papers; (10) Top 5% of Accepted (seminal paper).
- **Reviewer confidence**: The reviewer's level of confidence in the paper's assessment. The ICLR conference gives the following meaning to each confidence levels: (1) The reviewer's evaluation is an educated guess; (2) The reviewer is willing to defend the evaluation, but it is quite likely that the reviewer did not understand central parts of the paper; (3) The reviewer is fairly confident that the evaluation is correct; (4) The reviewer is confident but not absolutely certain that the evaluation is correct; (5) The reviewer is absolutely certain that the evaluation is correct and very familiar with the relevant literature.

For training the models, only the review text and the score given after the rebuttal phase were used.

From the metadata, it is possible to identify the old and updated version of the reviews' texts, and also the score and confidences change in each phase (*pre-rebuttal* and *post-rebuttal*). To identify the text change and extract the snippets added in the *post-rebuttal* phase, we use the python *difflib*[1] library, which allows comparing two strings and returning what's different between them. To identify the change in score and in confidence, it was only necessary to verify whether the score or confidence was different in the two states. From this, we classify each review according to the logic expressed in Chapter 4.2.

5.1.1 Dataset Characterization

Scores: As already mentioned previously, in our database reviews are scored between 1 and 10. Figure 5.1 shows the number of reviews that received each scores after the rebuttal phase. Note that the most frequent final score in the conference were 6 and 5, together they represent 48.19% of the database.

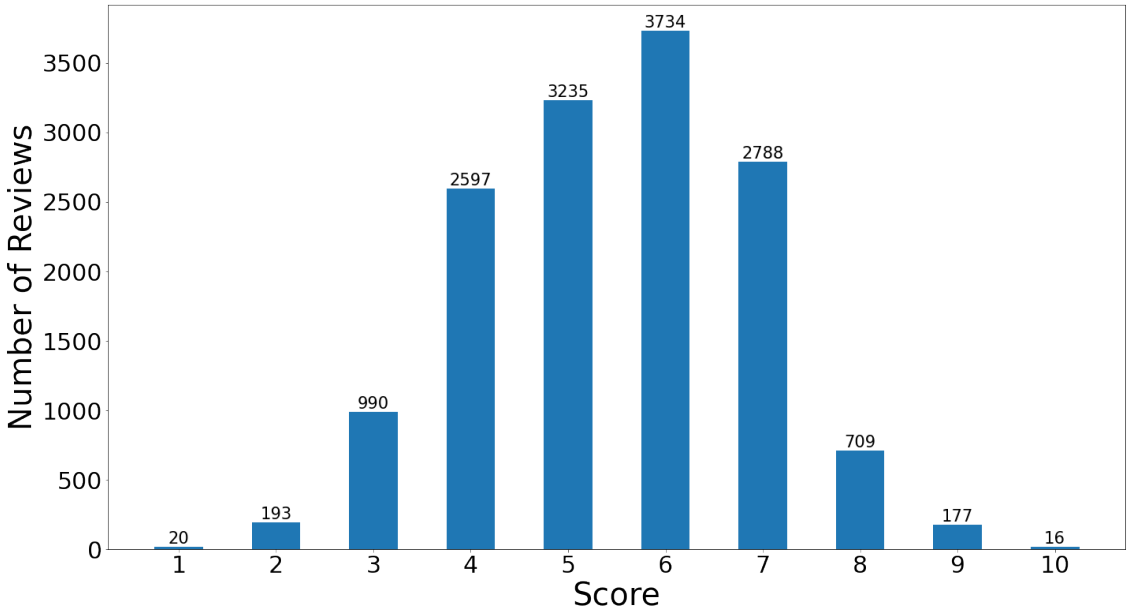


Figure 5.1: Number of reviews per score.

[1]https://docs.python.org/3/library/difflib.html

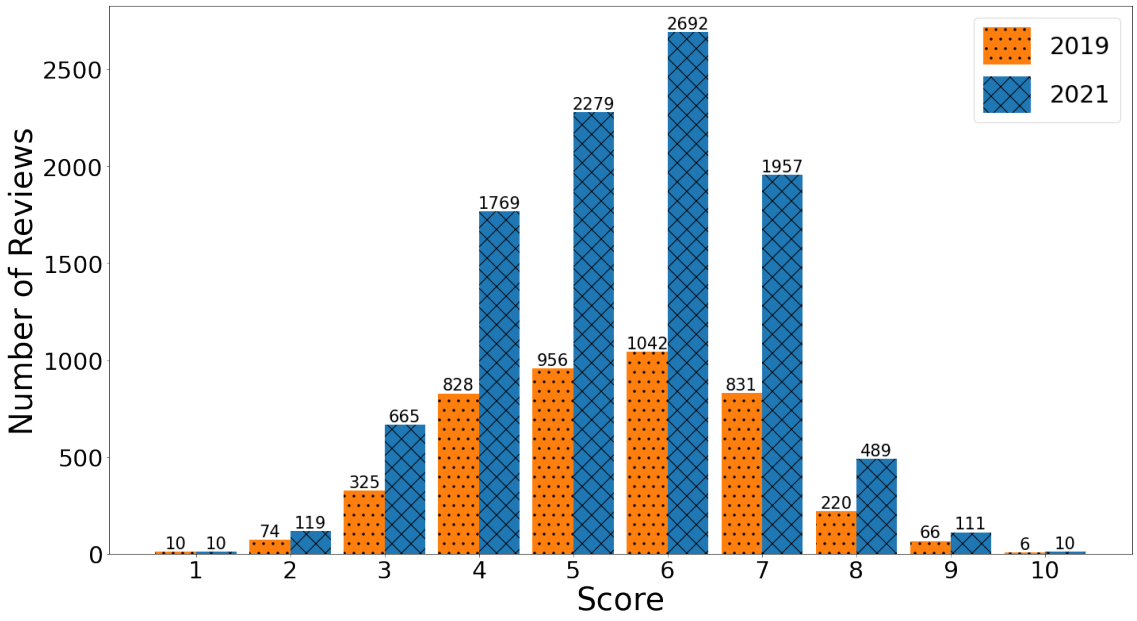


Figure 5.2: Number of reviews per Score, distinguishing between 2019 and 2021 reviews.

Figure 5.2 shows the number of reviews per score, but this time making a distinction between the reviews given to 2019 and 2021 papers. Observe that in the years 2019 and 2021, the order of frequency of the scores is the same, with score 6 being the most frequent and score 10 the least frequent.

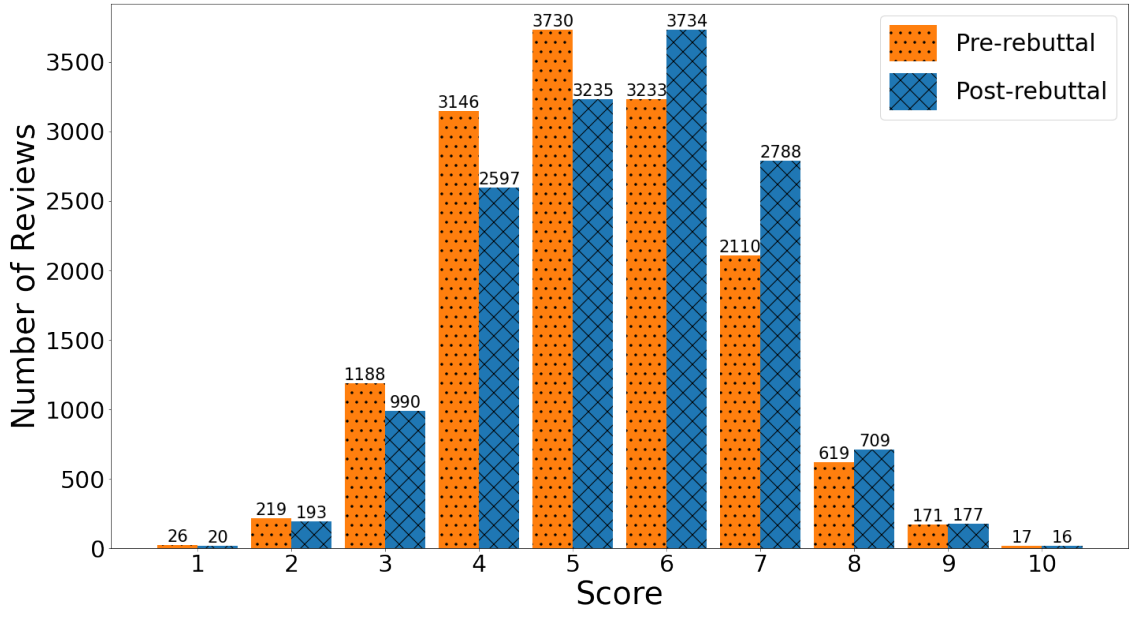


Figure 5.3: Number of reviews per score, distinguishing between *pre-rebuttal* and *post-rebuttal* scores.

Figure 5.3 shows the amount of reviews that received each score, but this time making a distinction between *pre-rebuttal* and *post-rebuttal* scores given. Note that 5 and 6 were the most given scores by reviewers, and that together they already represented 48.15% of the dataset. It is also possible to notice that the number of reviews with scores on the rejection side (<= 5) tends to decrease from *pre-rebuttal* phase to *post-rebuttal* phase, while the number of reviews with scores that indicate acceptance (>= 6), with the exception of the score 10, tends to increase. The change of scores after the rebuttal makes the score 7, the fourth most frequent in the *pre-rebuttal* phase, become the third most frequent score in the *post-rebuttal* phase, changing its position with the score 4, before the third most frequent, and in the end, the fourth most frequent.

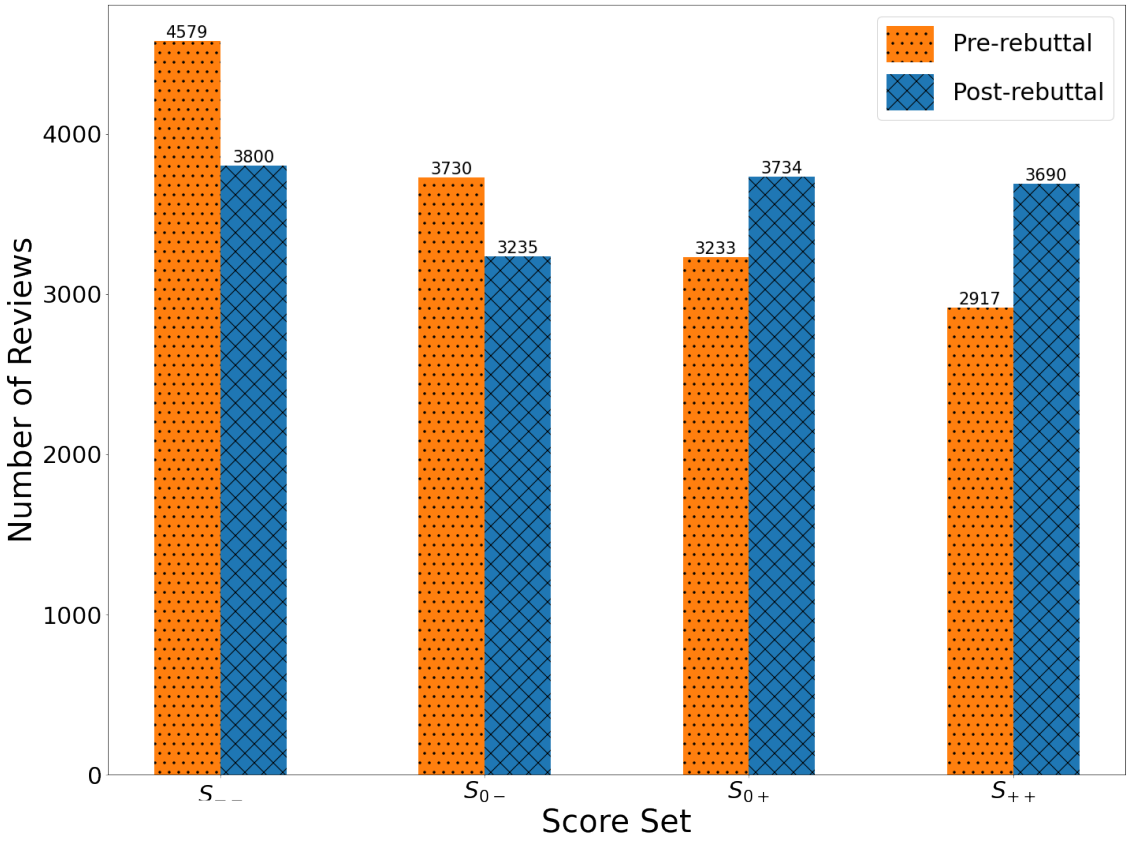


Figure 5.4: Number of reviews per score set, distinguishing between *pre-rebuttal* and *post-rebuttal* scores.

Figure 5.4 shows the number of reviews per score set, that is, with the scores grouped according to the logic described in Section 4.1.1. Note that with the scores grouped, the class with the highest amount of reviews is S_{--}. Note also that this grouping makes the classes have a more balanced amount. Figure 5.5 shows the score changes that occurred between the *pre-rebuttal* and *post-rebuttal* phases. Note that among 14,459 reviews, 2,766 underwent class changes. Of those modified, 2,469 (89.26%) moved to a higher score class and 297 (10.74%) moved to a lower class. Futhermore, observe that

1,420 (51.33%) scores changed from a reject score to an accept score, while 146 (5.27%) scores changed from accepted score to a rejected score.

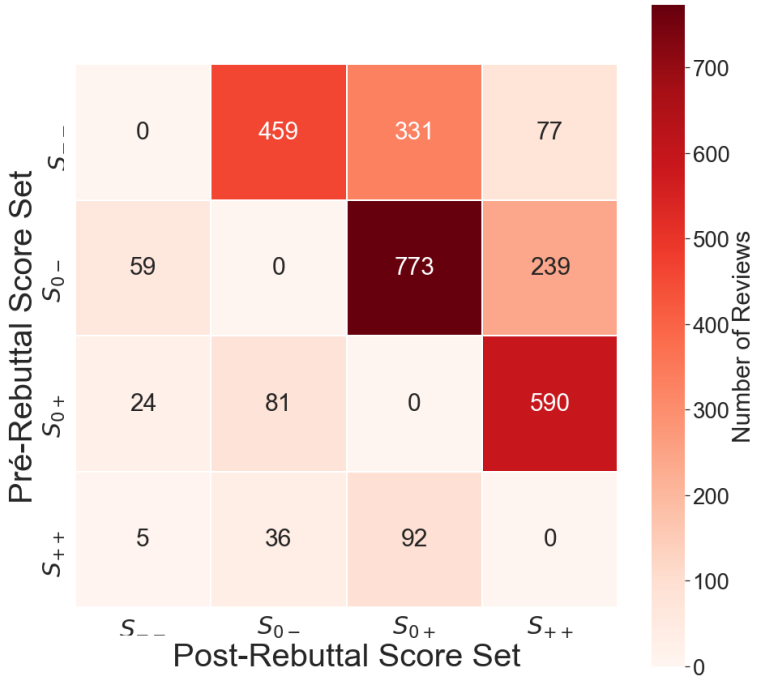


Figure 5.5: Number of reviews score set changes between the *pre-rebuttal* and *post-rebuttal* phases

Reviewers Confidence: Figure 5.6 shows the number of reviews per level of confidence. Note that the most frequent confidence is 4, with 4,880 reviews (48.73%).

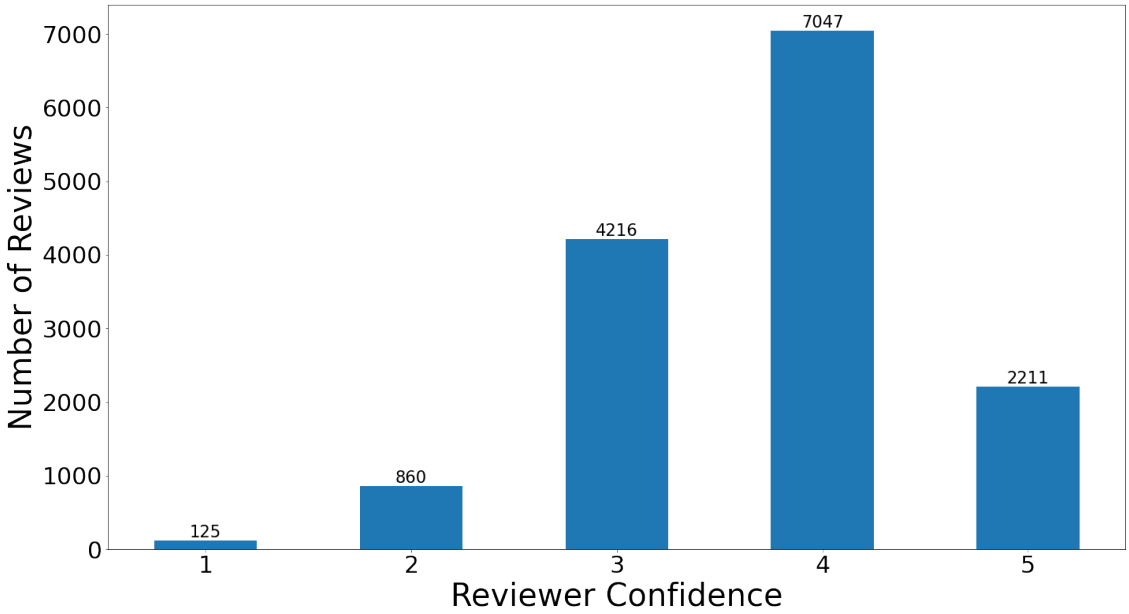


Figure 5.6: Number of reviews per reviewers confidence

This confidence means that the reviewers are confident, but not completely sure about the evaluation. The least frequently confidence is 1 with 125 reviews, meaning that only 0.86% of reviews were written as a guess. Overall, few reviewers from the conference and years analyzed gave their scores with low confidence.

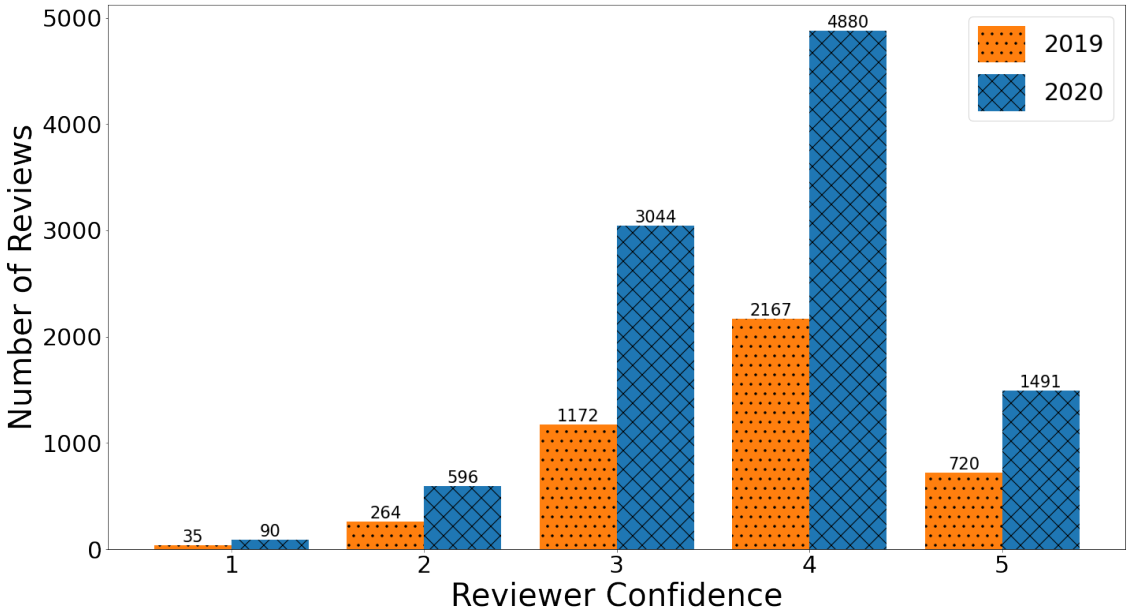


Figure 5.7: Number of reviews per reviewers confidence, distinguishing between 2019 and 2021 reviews.

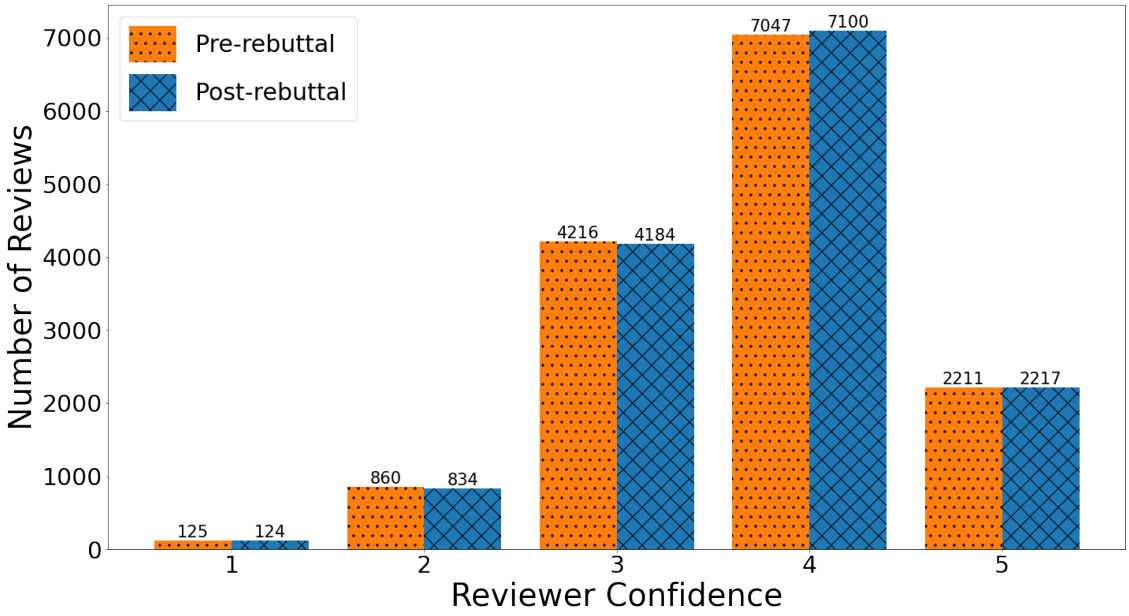


Figure 5.8: Number of reviews per reviwers confidence, distinguishing between *pre-rebuttal* and *post-rebuttal* confidence.

The Figure 5.7 shows the number of reviews per reviewer confidence level, but this time distinguishing between the years of the ICLR conference. Note that in the years

2019 and 2021, the order of confidences frequency is the same, with confidence 4 being the most frequent and confidence 1 being the least frequent. The Figure 5.8 shows the number of reviews that each reviewer confidence classes has, distinguishing between the *pre-rebuttal* and *post-rebuttal* phases. Note that unlike the scores, reviewers do not change their confidence much, after the rebuttal phase. Only 362 (2.50%) of the reviews had their confidence changed.

Figure 5.9 shows the changes in reviwer confidence classes that occurred between the pre-rebuttal and post rebuttal phases. Note that of the reviews with altered confidence, 229 (63.26%) the confidence of frequency, while 133 (36.74%) it decreased.

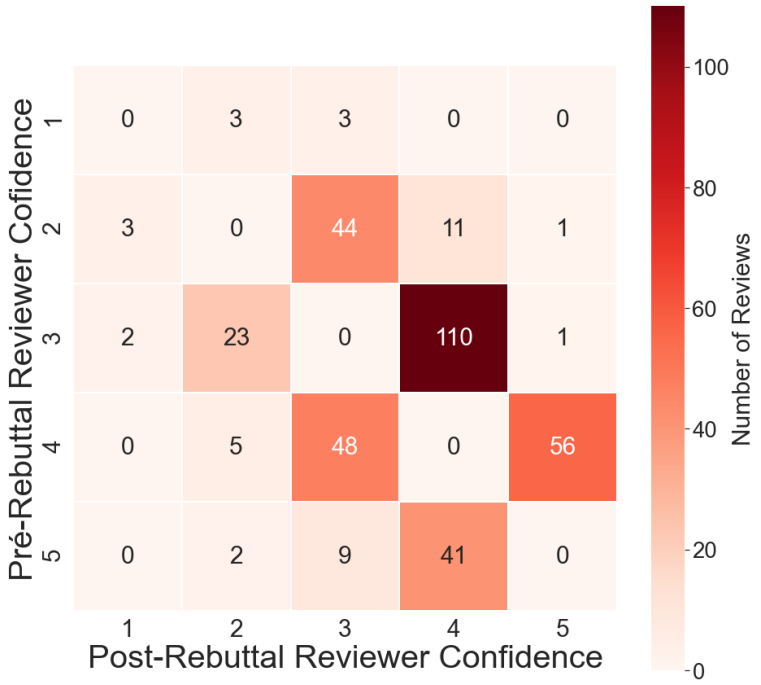


Figure 5.9: Number of reviwers confidence changes between the *pre-rebuttal* and *post-rebuttal* phases.

Scores Set x Reviewers Confidence: Figure 5.10 shows the number of reviews in the relationship between the score set and the reviewer's confidence. The colors indicate the magnitude of the values, that way, the closer to red, the lower the quantty of reviews, and the closer to green, the higher the quantity of reviews.

Figure 5.11 shows the probability of the reviewers confidence, given the score that she/he gave. The colors indicate the magnitude of the values, that way, the closer to red, the lower the probability, and the closer to green, the higher the probability. Note that regardless of the score, reviewers have a high probability of giving them with high confidence.

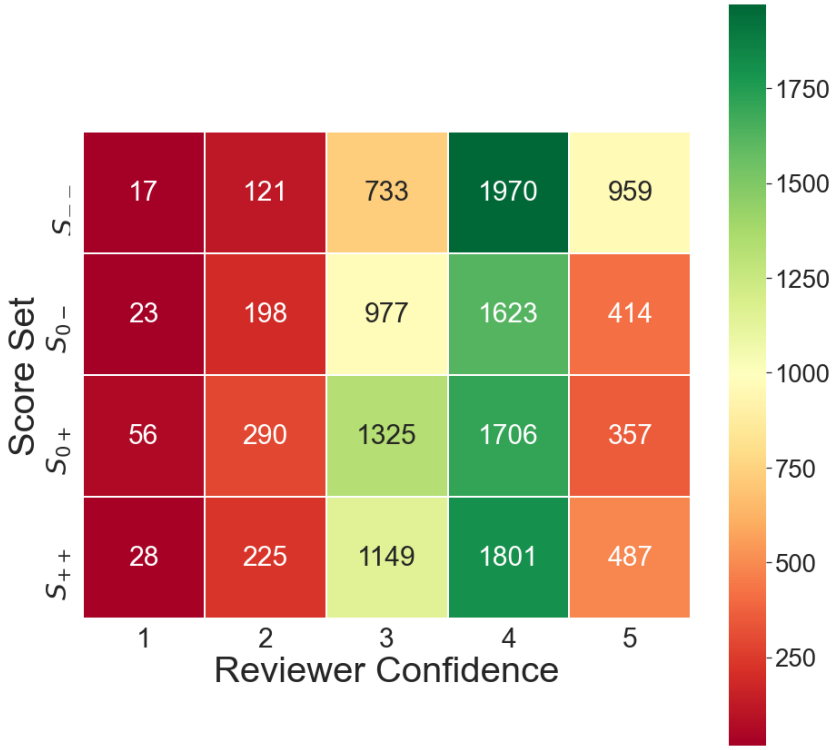


Figure 5.10: Number of reviews considering the score set and the reviwer confidence.

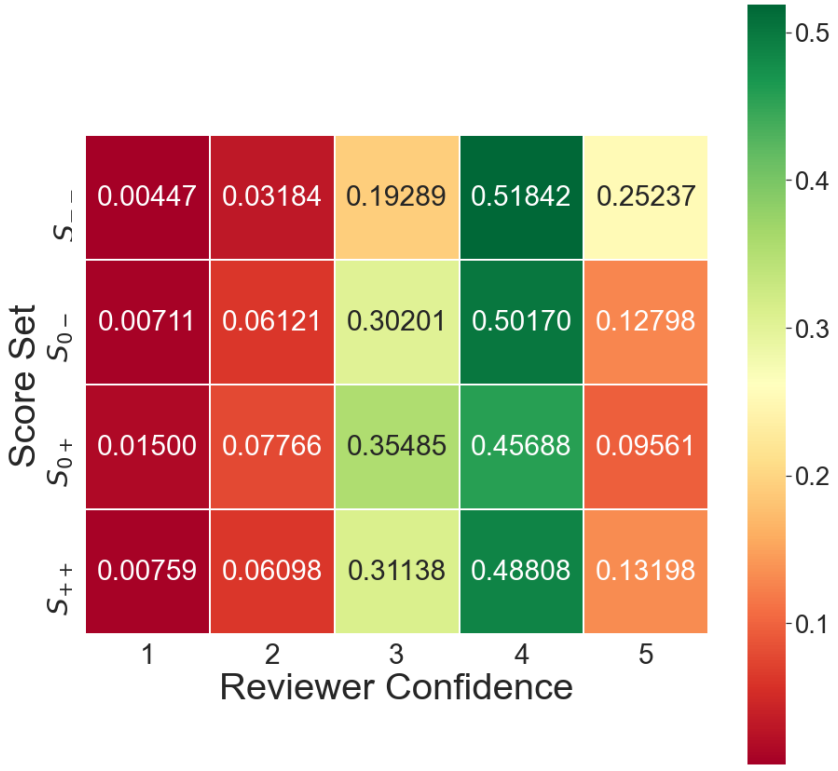


Figure 5.11: Probability of the reviewer confidence based on the review score set.

Number of Words x Score Set: Figure 5.12 shows the number of words in *pre-rebuttal* text review, distinguishing between the score set given before the rebuttal. Figure 5.13

shows the number of words in the reviewers' answer to authors after the rebuttal phase. At both figures, the information is represented by violin plots, in which, the wider the violin body is on the *Number of Word* axis, the greater the number of reviews. The Figures also bring the first, second and third quartile of the number of words in each of the score set. Note that there is a tendency for reviewers to write more when the score is lower, this behavior can be explained by the fact that it is necessary to write bigger justifications for lower score.

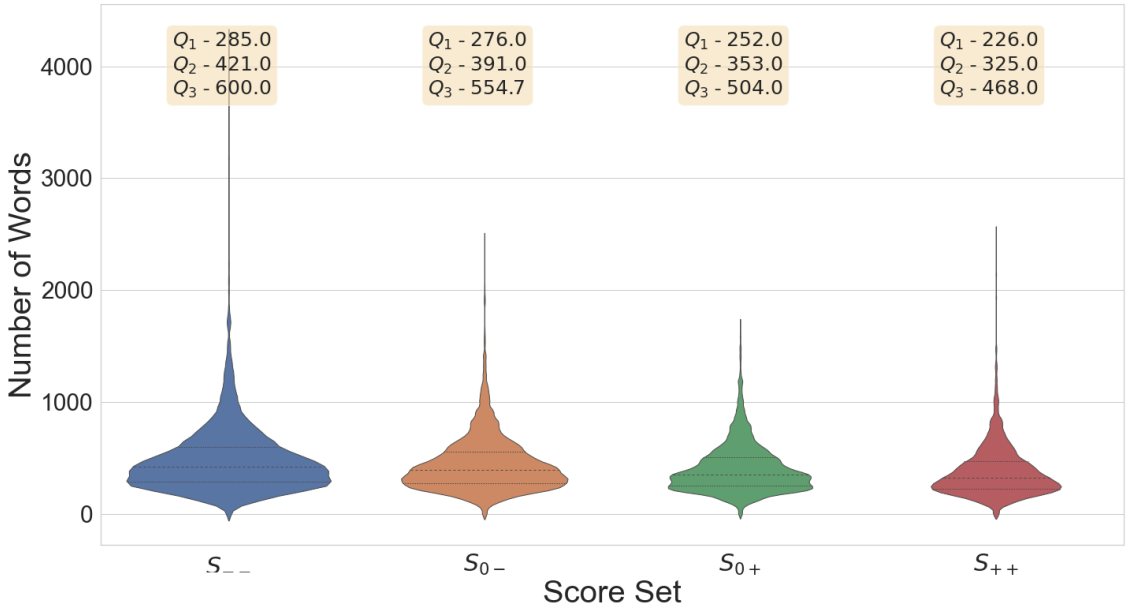


Figure 5.12: Number of words in *pre-rebuttal* review text per score set.

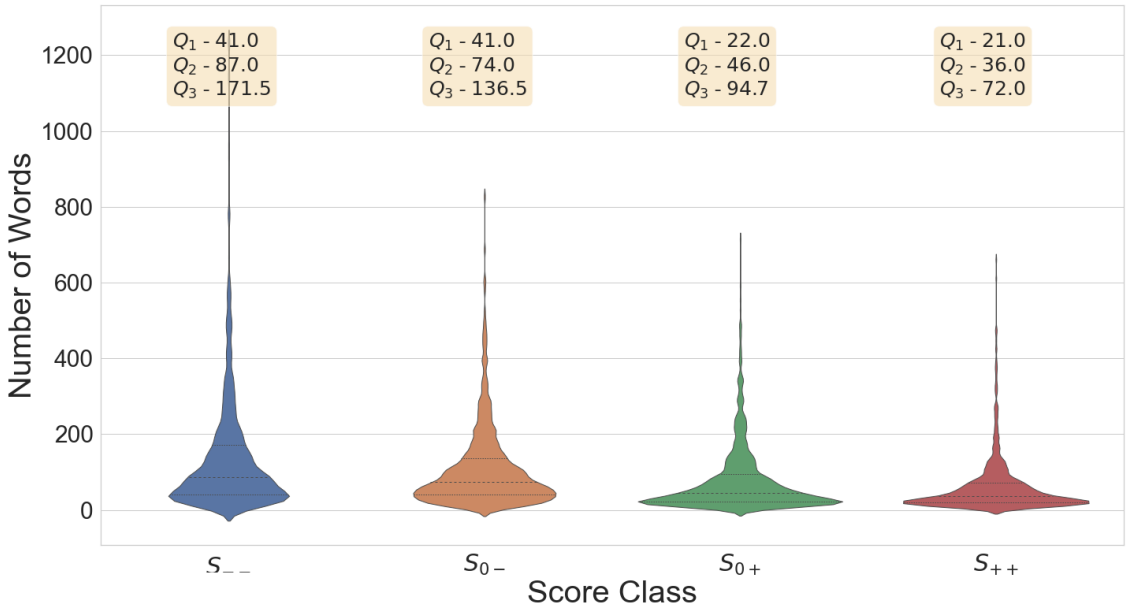


Figure 5.13: Number of words in *post-rebuttal* review complement text per score set.

Number of Words x Reviewers Confidence: Figure 5.14 shows the number of words in the *pre-rebuttal* review text, distinguishing between the reviwer confidence level given in *pre-rebuttal* phase. Figure 5.15 shows the number of words in the reviewers' response to authors after the rebuttal phase. Both also bring the first, second and third quartile of the number of words in each of the classes. Note that there is a tendency that reviewers with low confidence tend to write less before and after the rebuttal phase.

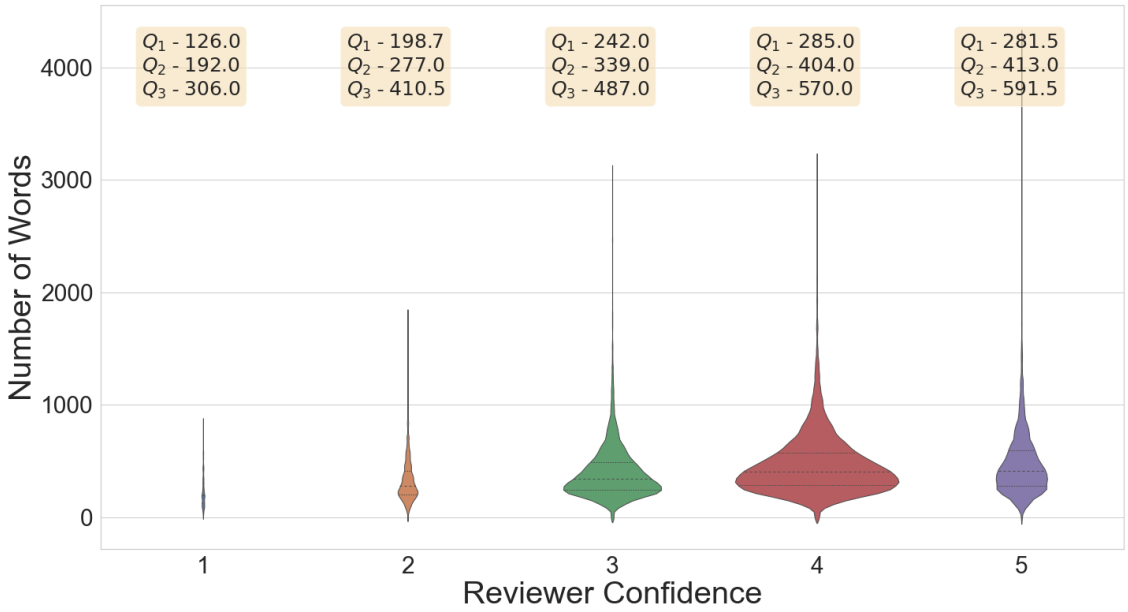


Figure 5.14: Number of words in *pre-rebuttal* review text per reviewer confidence.

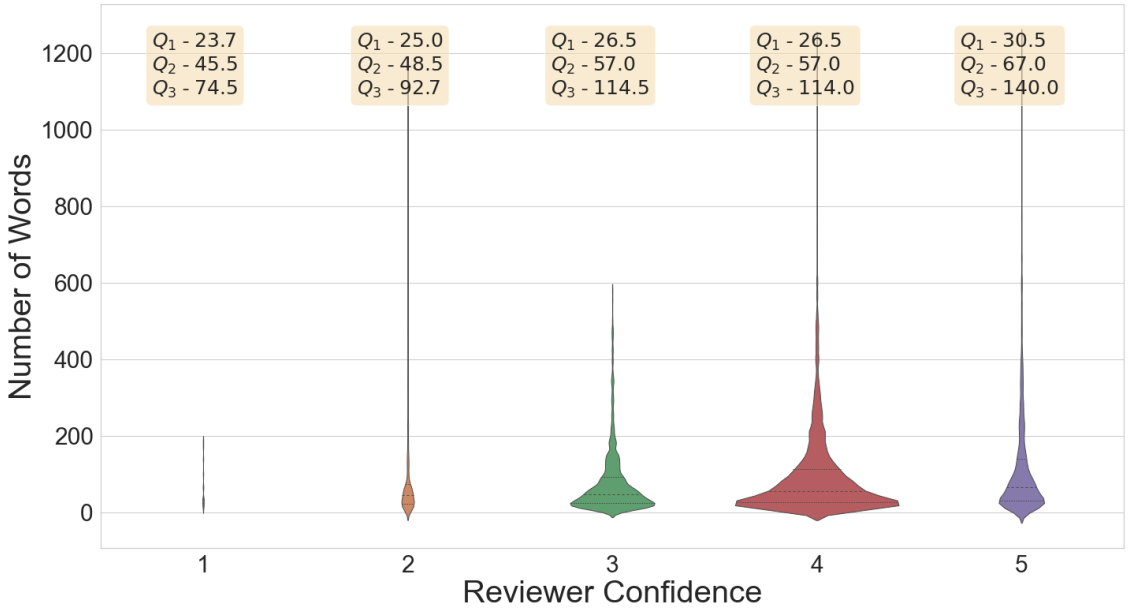


Figure 5.15: Number of words in *post-rebuttal* review complement text per reviewer confidence.

5.1.2 Dataset Distribution for the Experiments

In Table 5.1 we describe the amount of instances of each class for both tasks evaluated in this work. For the *review score prediction* task, the S_{--} group contains 4,144 reviews, the S_{0-} group contains 3,441 reviews, the S_{0+} group contains 3,530 reviews and the S_{++} group contains 3,344 reviews, a roughly balanced set in term of review scores. For *paper decision prediction* task, there are 1,362 accepted papers and 2,673 rejected papers, slightly unbalanced towards the rejection class, as expected.

Data distribution for Model Comparison			
Review Score Prediction (*RSP*)			
S_{--}	S_{0-}	S_{0+}	S_{++}
4,144	3,441	3,530	3,344
Paper Decision Prediction (*PDP*)			
Accepted		*Rejected*	
1,362		2,673	

Table 5.1: Data distribution to analyze the overall performance of the models in the tasks of *review score prediction* and *paper decision prediction*.

In Table 5.2 we describe the amount of *easy instances* and *hard instances* found in the data regarding the state of the review. For the *RSP* task, there are 10,134 *easy instances* (S_0T_0), 3,022 *hard instances* ($S_0T_1 \cup S_1T_1$), and 1,303 reviews of the *hardest instance* (S_1T_0) class. For the *PDP* task, there are 1,501 *easy instances* ($\mathcal{S}_0\mathcal{T}_0$), 1,482 *hard instances* ($\mathcal{S}_*\mathcal{T}_1$), and 1,052 of the *hardest instance* ($\mathcal{S}_1\mathcal{T}_0$) class. Note that the amount of *hard instances* is significant, which can be a challenge for classifiers.

Data distribution for Review and Paper Classes			
	Easy	*Hard*	*Hardest*
Review Score Prediction (RSP)	10,134	3,022	1,303
Paper Decision Prediction (PDP)	1,501	1,482	1,052

Table 5.2: Data distribution to analyze the performance of models in the tasks of *RSP* and *PDP* when exposed to hard instances related to score and/or text change.

Finally, in Table 5.3 we show the amount of *hard instances* regarding the scores of the review. In order to isolate the impact of the review score from the review state, for both tasks we only consider reviews that have not gone through any change in the process (*easy instances*). For the *review score prediction* task, the S_{--} group contains 3,153 reviews, the S_{0-} group contains 2,325 reviews, the S_{0+} group contains 2,269 reviews and the S_{++} group contains 2,387 reviews. For *paper decision prediction* task, there are 324 accepted papers and 1,177 rejected papers.

Data distribution for Borderline Reviews			
Review Score Prediction (*RSP*)			
S_{--}	S_{0-}	S_{0+}	S_{++}
3,153	2,325	2,269	2,387
Paper Decision Prediction (*PDP*)			
Accepted		*Rejected*	
324		1,177	

Table 5.3: Data distribution to analyze the performance of models when exposed to hard instances related to borderlines.

5.2 Models

In this work we will evaluate several state-of-the-art models for the *review score prediction* (*RSP*) and *paper decision prediction* (*PDP*) tasks. The first, *DeepSentiPeer* [Ghosal et al., 2019a], to perform the score and and the final decision prediction tasks, it was trained only with the review texts (one of the three approaches proposed by the authors), as shown in Figure 5.16 and Figure 5.17, to maintain consistency with the other models evaluated in this work. Furthermore, unlike what is done in Ghosal et al. [2019a], where the model is evaluated in a regression task, here it will be evaluated in a classification task. Although the distance between borderline rejection instances (S_{0-}) and acceptance borderline (S_{0+}) is small, such evaluations represent different spectrums in the context of the final decision of the paper and wrong classifications of these scores should be penalized with due rigor.

The second, *HabNet* [Deng et al., 2020], is trained with the texts of the reviews to predict the score (Figura 5.16) and the final decision (Figura 5.17), and already originally

treat these tasks as classification tasks. The model was initialized with pre-trained GloVe embeddings following the authors' guidelines.

Figure 5.16: Approach used for the task of predicting the score of a paper. The model is trained on the review text, and predict the score class.

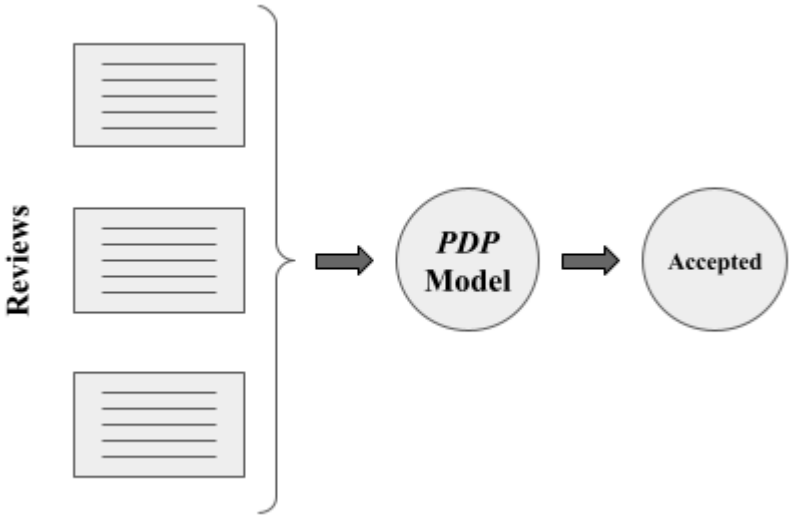


Figure 5.17: Approach used for the task of predicting the final decision of a paper. (*DeepSentiPeer* and *HabNet*). The model is trained on the text of all reviews of a paper, and predict the final decision.

We also evaluated other state-of-the-art text classification models, such as *C-LSTM* [Zhou et al., 2015] and *CNN-GRU* [Wang et al., 2016], which were initialized with pre-trained word2vec vectors generated from within Google News data set. *BERT* [Devlin et al., 2019], *RoBERTa* [Liu et al., 2019] and *XLNet* [Yang et al., 2019] were pre-trained with texts from Wikipedia and BookCorpus. For all these models, the prediction of the score (*RSP*) is made from the review texts (Figure 5.16). For the prediction of the final decision (*PDP*), we used a different approach from the ones proposed by the *DeepSentiPeer* and *HabNet* models. Instead of using the review text, we use the output of the trained model for the *RSP* task. In other words, we train the model normally for the task *RSP*, then, for each review r_i, we collect the probability given by the model of the text t_i corresponding to a score belonging to the groups S_{--} ($P_{--}(t_i)$), S_{0-} ($P_{0-}(t_i)$),

S_{0+} ($P_{0+}(t_i)$) and S_{++} ($P_{++}(t_i)$). Then, for each paper p_i, we group all the calculated probabilities for all reviews $r_i \in R_i$ into a single matrix, which serves as input to a layer (Multilayer Perceptron) responsible for training and predicting the final decisions.

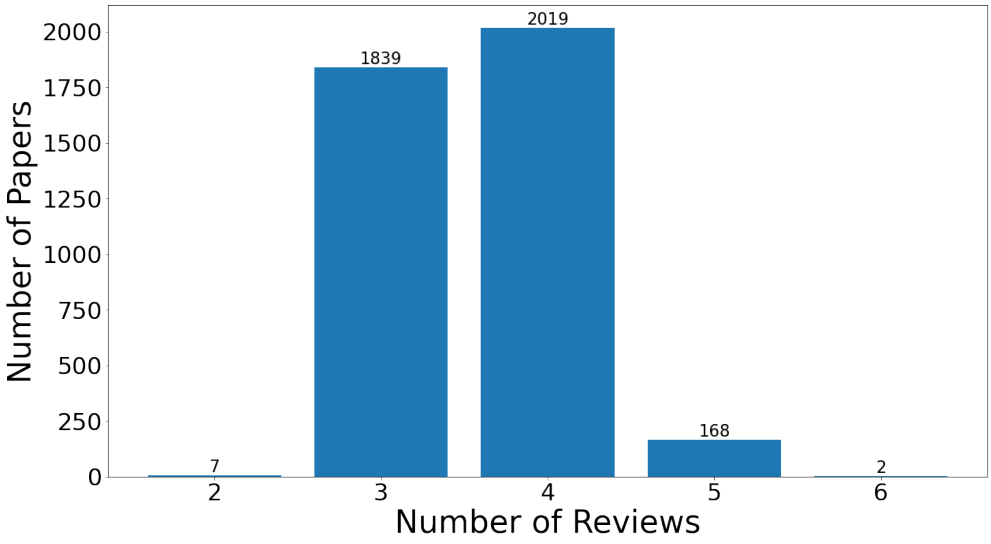


Figure 5.18: Number of papers per number of reviews.

As the number of reviews ($|R_i|$) per paper varies between 2 and 6 (Figure 5.18), the size of the probability vectors also varies between 8 and 24. To make the input vector for the MLP a fixed size, we define its size as 24 and we did a *average padding* to fill in the missing values. In other words, for all papers with less than 6 reviews, we take the (column-wise) average among the probability vectors and create new vectors with these averages until completing 6 vectors. Thus, all papers are associated with inputs of the same size, that is, vectors of size 24. We tried other ways to handle these differences, but the results were similar or worse (Appendix A).

Figure 5.19 illustrates the approach used. Each one of the reviews associated to papers is submitted to a *review score prediction* (*RSP*) model (*C-LSTM*, *CNN-GRU*, *XLNet*, *RoBERTa*, or *BERT*) trained to predicted the score class (S_{--}, S_{0-}, S_{0+}, and S_{++}) of a review from the text written by the reviewer. From these trained models, we extracted the probability of each reviews belong to one of possibles score class (P_{--}, P_{0-}, P_{0+}, and P_{++}). A new vector is created containing the average of the probabilities of the vectors generated by the *RSP* models. Then, all the vectors are concatenated to create a matrix. The vector that contains the averages probabilities is repeated in the matrix util it has six rows, that is the maximum amount of reviews that a paper can have in our database. The matrix is used as input to train of the MLP model, whose goal is to predict the final paper decision.

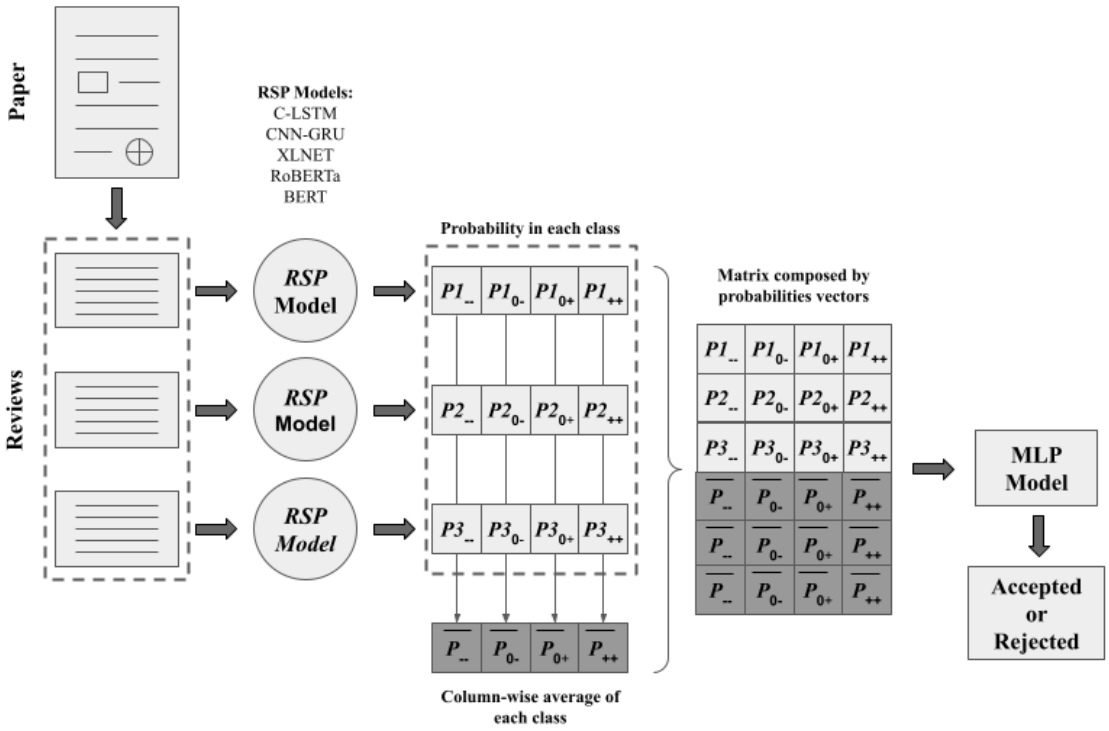


Figure 5.19: New approach used for the task of predicting the final decision of a paper.

Chapter 6

Experimental Results

6.1 Model Comparison and Overall Results

In order to understand the feasibility of an automatic review classification system, our first step is to assess the overall performance of the models. For this, we performed a 5-fold cross validation and calculated the mean and standard deviation of the accuracy. To assess the statistical significance of the results, we ran the t-test to determine if there was a significant difference between the means.

Note in Table 6.1 that in the *review score prediction* (*RSP*) task, the models *HabNet* and *BERT* performed best. The models *RoBERTa*, *DeepSentiPeer* and *XLNet* were statistically tied and had the second best performance. Finally, *C-LSTM* and *CNN-GRU* had the worst results, having a performance 50% lower than the best models. It is important to highlight that among *HabNet* and *DeepSentiPeer*, created specifically for the studied task, *DeepSentiPeer* had a statistically worse performance, being also behind *BERT*.

Observe in Table 6.1 that, for the *paper decision prediction* (*PDP*) task, *BERT*, *DeepSentiPeer* and *HabNet* performed best, being statistically tied. *RoBERTa* and *XLNet* models were statistically tied with the second best performance. Finally, *C-LSTM* and *CNN-GRU* were tied and presented the worst performance, with a performance of 14% lower. It is important to emphasize that the models created specifically for the studied task, *HabNet* and *DeepSentiPeer*, had a similar performance to *BERT*.

Regarding the results, for the *RSP* task, the best models had an average accuracy of $\approx 50\%$, while for the *PDP*, they had an average accuracy of $\approx 75\%$. These results suggest that we are still far away from a completely autonomous peer review system, both in terms of scoring the reviews and deciding the acceptance of a paper. Also note that the proposal to use the output of the trained model for the *RSP* task in the *PDP* task had a similar performance when compared to the strategy used by the *DeepSentiPeer* and *HabNet* models, which use only the text of the reviews to predict the final decision. Finally, observe that there is an expected correlation between the two tasks, that is, the

best (worst) models in the *RSP* task are also the best (worst) models in the *PDP* task.

Overall Results		
	Review Score Prediction (*RSP*)	**Paper Decision Prediction (*PDP*)**
Model	**Input: Text**	**Input: Text**
DeepSentiPeer	45.72% (+/- 0.95%)	**75.79% (+/- 0.86%)**
HabNet	**49.51% (+/- 0.92%)**	**75.39% (+/- 0.65%)**
Model	**Input: Text**	**Input: *RSP* Output**
C-LSTM	24.81% (+/- 1.28%)	66.25% (+/- 0.06%)
RNN-GRU	25.03% (+/- 0.67%)	66.25% (+/- 0.06%)
XLNet	44.26% (+/- 0.94%)	73.88% (+/- 0.92%)
RoBERTa	45.93% (+/- 1.08%)	74.85% (+/- 0.65%)
BERT	**49.20% (+/- 1.00%)**	**76.88% (+/- 0.99%)**

Table 6.1: Accuracy of models in *RSP* and *PDP* tasks.

6.2 Impact of Changes After the Rebuttal

Having assessed the overall performance of the classification models on both tasks, our next step is to see how they deal with the first type of *hard instances*, those related to review and paper states throughout the process. For that, we will use the two models that had the best performance in the previous tasks, *HabNet* and *BERT*. Furthermore, this choice will also allow us to evaluate the performance of two different training approaches in the *PDP* task. While *HabNet* is trained only with reviews texts, *BERT* is trained with the outputs of the model trained for the *RSP* task, that is, it is indirectly trained with the predicted scores (Figure 5.19).

6.2.1 Scenarios

To assess the impact of *hard instances* on model training, three different training scenarios were considered for each task. For the task *RSP*, in the first scenario, we train the models only with reviews from the group S_0T_0 (*easy instances*), in the second, with reviews from the group $S_0T_0 \cup S_*T_1$ (*easy instances* + *hard instances*) and, in the third (*easy instances* + *hard instances* + *hardest instances*), using all reviews. For the *PDP* task, in the first scenario we train the models only with papers from the group $\mathcal{S}_0\mathcal{T}_0$ (*easy instances*), in the second, with papers from the group $\mathcal{S}_0\mathcal{T}_0 \cup \mathcal{S}_*\mathcal{T}_1$ (*easy instances* + *hard instances*) and, in the third, with all papers (*easy instances* + *hard instances* + *hardest instances*). Afterwards, each trained model was tested separately in groups of *easy instances*, *hard instances* and *hardest instances* corresponding to each task.

6.2.2 Results

Note in Table 6.2 that, for the *review score prediction* task, the *HabNet* model performed better than *BERT* to predict instances of the S_*T_1 group, regardless of the

Review Score Prediction (*RSP*)					
		Test			
	Scenario	Easy	Hard	Hardest	Hard + Hardest
Train	Easy	**52.44%** ♠ **(+/- 0.63%)**	40.44% ♠ (+/- 1.39%)	43.36% ♠ (+/- 1.88%)	41.32% ♠ (+/- 0.79%)
		52.06% ♢ **(+/- 0.40%)**	43.18% ♢ (+/- 1.23%)	45.90% ♢ (+/- 2.44%)	42.04% ♢ (+/- 1.32%)
	Easy + Hard	**51.60%** ♠ **(+/- 1.11%)**	42.92% ♠ (+/- 1.43%)	41.52% ♠ (+/- 3.28%)	42.50% ♠ (+/- 1.27%)
		51.61% ♢ **(+/- 0.63%)**	47.22% ♢ (+/- 0.83%)	43.74% ♢ (+/- 2.76%)	44.67% ♢ (+/- 1.48%)
	Easy + Hard + Hardest	**52.02%** ♠ **(+/- 0.71%)**	42.62% ♠ (+/- 1.09%)	44.74% ♠ (+/- 2.14%)	43.26% ♠ (+/- 1.13%)
		51.57% ♢ **(+/- 0.79%)**	46.59% ♢ (+/- 0.75%)	45.28% ♢ (+/- 2.34%)	44.35% ♢ (+/- 1.43%)

Table 6.2: ♠ *BERT* and ♢ *HabNet*. Accuracy in *review score prediction* task.

training scenario. For the other groups, the performance was similar. The addition of

hard instances and *hardest instances* to training impacted the prediction of instances of the S_*T_1 group by the *HabNet* model. In this case, surprisingly, the more difficult the instances, the better the model's performance. For the other cases, the addition of these instances in training did not change the performance of the models. Regardless of the training model and scenario, the prediction of instances of the S_0T_0 group was, as expected, better than the instances of the S_*T_1 and S_1T_0 group. The biggest difference between the worst and the best case, within the same training scenario, was 12% (Model: *BERT*, Training: *easy instances*, Test: *easy instances*, Test: *hardest instances*).

Paper Decision Prediction (*PDP*)					
		Test			
	Scenario	Easy	Hard	Hardest	Hard + Hardest
Train	Easy	**84.81% ♠ (+/- 1.37%)**	74.83% ♠ (+/- 1.11%)	61.50% ♠ (+/- 1.48%)	69.30% ♠ (+/- 0.91%)
		83.54% ♢ (+/- 1.21%)	72.33% ♢ (+/- 1.10%)	67.21% ♢ (+/- 3.85%)	68.55% ♢ (+/- 1.57%)
	Easy + Hard	**86.74% ♠ (+/- 1.62%)**	75.84% ♠ (+/- 2.76%)	65.40% ♠ (+/- 3.54%)	70.84% ♠ (+/- 2.77%)
		84.28% ♢ (+/- 0.75%)	75.24% ♢ (+/- 1.14%)	67.78% ♢ (+/- 2.69%)	71.31% ♢ (+/- 1.47%)
	Easy + Hard + Hardest	**85.81% ♠ (+/- 2.05%)**	75.98% ♠ (+/- 2.35%)	65.78% ♠ (+/- 2.06%)	72.93% ♠ (+/- 1.64%)
		83.48% ♢ (+/- 1.40%)	75.84% ♢ (+/- 1.29%)	67.68% ♢ (+/- 2.31%)	71.67% ♢ (+/- 1.71%)

Table 6.3: ♠ *BERT* and ♢ *HabNet*. Accuracy in *paper decision prediction* task.

For the *paper decision prediction* task, observe in Table 6.3 that *HabNet* performed better than *BERT* to predict instances of the $\mathcal{S}_1\mathcal{T}_0$ group when trained with the $\mathcal{S}_0\mathcal{T}_0$ group. For the other groups, the performance was similar. The addition of more difficult instances to training impacted the prediction of instances of the group $\mathcal{S}_*\mathcal{T}_1$ and $\mathcal{S}_1\mathcal{T}_0$ by *HabNet* and *BERT* respectively. In these cases, as for the *RSP* task, surprisingly, the more difficult the instances, the better the model's performance. For the other cases, the addition of these instances to the training did not change the performance of the models. As with the *RSP* task, regardless of the model and training scenario, the prediction of instances of the group $\mathcal{S}_0\mathcal{T}_0$ was better than the instances of the group $\mathcal{S}_*\mathcal{T}_1$ and $\mathcal{S}_1\mathcal{T}_0$. The biggest difference between the worst and the best case, within the same training scenario, was 23.31% (Model: *BERT* Training: *easy instances*, Test: *easy instances*, Test: *hardest instances*).

In short, the results in this section suggest that reviews that went through changes in the process (*hard instances*) are the hardest to classify. We also observed that adding

such instances in training does not harm their performance in *easy instances*, but improve their performance in *hard instances* (*RSP*) and in all classes (*PDP*).

6.3 Impact of Bordeline Reviews

In this section we investigate the impact of *hard instances* related to borderline scores. For this, we will use only *BERT*, as in addition to having obtained the best results in both tasks, it also allows for an additional experiment to be carried out. In order to investigate whether it is really necessary to impose an acceptance decision on all review scores, we will evaluate *BERT* on the *PDP* task when it is trained with only three classes of scores (S_{--}, S_{0*} and S_{++}) instead of four (S_{--}, S_{0-}, S_{0+}, and S_{++}).

6.3.1 Results

First, in Table 6.4, we show how *BERT* performs in binary classification tasks, where the goal is to tell apart one group (e.g. S_{--}) from another (e.g. S_{++}). As expected, when *BERT* is working with the easiest classes (S_{--} and S_{++}), the performance is 55.53% greater than when it tries to separate the most difficult instances ($S_{0-}$and S_{0+}), i.e., a difference in accuracy of 31.43%. Also, note how borderline instances degrade the classifier performance.

BERT One vs One		
S_{--} ***vs*** S_{0-}	S_{--} **vs** S_{0+}	S_{0-} **vs** S_{0+}
63.36% (+/- 1.49%)	78.26% (+/- 1.41%)	56.60% (+/- 1.19%)
S_{++} **vs** S_{0+}	S_{++} **vs** S_{0-}	S_{--} **vs** S_{++}
62.24% (+/- 1.10%)	78.65% (+/- 1.33%)	**88.03% (+/- 1.01%)**

Table 6.4: Accuracy in the *review score prediction* (*RSP*) task. Training and testing with just two classes.

When the classifier have to separate instances with opposite decisions (accept/reject) and

one of them is a borderline (eg. S_{++} and S_{0-}), the performance is 10% lower than between instances with easy opposite decisions and no borderline score (S_{--} and S_{++}).

We now evaluate whether *BERT* trained with a single neutral score (S_{0*}) instead of two (S_{0-} and S_{0+}) with harm its performance in the *PDP* task. Observe in Table 6.5 that grouping borderlines into a single class S_{0*} caused a drop in accuracy when predicting *easy instances* ($\approx$ 85% to $\approx$ 81%). For the *hardest instances*, the accuracy was statistically the same. Surprisingly, for *hard instances*, grouping borderlines into a single class resulted in a performance improvement ($\approx$ 75% to $\approx$ 80%).

BERT - Paper Decision Prediction (PDP)					
		Test			
	Scenario	Easy	Hard	Hardest	Hard + Hardest
Train	Easy	**80.89% (+/- 1.65%)**	79.29%% (+/- 2.50%)	64.73% (+/- 3.30%)	72.53% (+/- 2.31%)

Table 6.5: Accuracy in the *PDP* task. Training on Easy instances. Test on Easy, Hard, Hardest and Hard + Hardest instances.

The results suggest that distinguishing the borderline classes from one another is a difficult task, however, dividing the borderline classes into two may not be necessary, as this does not make the models perform better than the scenario in which the borderlines are together.

Chapter 7

How far are we?

After analyzing the impact *hard instances* have on classifiers, we now seek to understand how far we are from being able to automate the process of scoring a paper solely based on the review text and deciding whether a paper should be accepted or rejected. For this, we measure the magnitude of the classifier error. At the time of classification, the classifier yields the probabilities of an instance to belong to each class. The class with the highest probability is chosen as the predicted class. We define the *magnitude of the classifier error* as the difference between the probability given to the incorrectly predicted class and the probability given to the correct class.

7.1 Review Score Prediction (*RSP*)

Table 7.1 shows the average confusion matrix for the *review score prediction* after five runs. As expected, the classifier tends to confuse neighboring classes, for example: $\approx$ 158 instances that should be predicted as S_{--} were predicted as S_{0-}; $\approx$ 124 instances that should be predicted as S_{++} were predicted as S_{0+}.

Correct Class x Predicted Class (*RSP*)					
		Predicted Class			
	Review Class	S_{--}	S_{0-}	S_{0+}	S_{++}
Correct Class	S_{--}	**387.2 (+/- 24.39)**	158.2 (+/- 13.64)	67.6 (+/- 13.64)	17.6 (+/- 3.97)
	S_{0-}	119.4 (+/- 12.75)	**184.8 (+/- 8.81)**	124.8 (+/- 9.94)	36 (+/- 7.37)
	S_{0+}	37.2 (+/- 7.13)	105.6 (+/- 11.37)	**184.8 (+/- 7.05)**	126.2 (+/- 12.27)
	S_{++}	12.6 (+/- 1.49)	34.4 (+/- 11.37)	124.4 (+/- 24.92)	**306 (+/- 27.87)**

Table 7.1: Confusion matrix of *RSP* task, with the average number of instances. This table refers to the scenario where the model was trained and tested in Easy instances.

Table 7.2 describes the average magnitude of the error when the classifier makes a mistake. The highest difference is 50.20%, when the correct class is S_{0-} and the classifier predicted S_{++}. This means that when such mistake occurs, the difference between the probability given by the classifier to the incorrect class (S_{++}) and the correct class (S_{0-}) is, on average, 0.502. In fact, note that when the classifier makes an error, the magnitude of the error is large, since in the best case, the average distance between the probabilities is 21%. Surprisingly, also note that the greater the distance between the score values, the greater the classifier error, that is, the greater the difference between the probability given to the correct class and the probability given to the incorrectly predicted class. This suggests that the classifier is fairly confident of these mistakes, which may characterize a *discrepant* instance, i.e., a review text that conveys the opposing polarity of its associated score.

Classifier Error - Correct Class x Predicted Class					
		Wrong Predicted Class			
	Review Class	S_{--}	S_{0-}	S_{0+}	S_{++}
Correct Class	S_{--}	*	21.18% (+/- 12.93%)	35.24% (+/- 13.17%)	**49.13% (+/- 17.39%)**
	S_{0-}	35.30% (+/- 24.01%)	*	25.40% (+/- 15.45%)	**50.20% (+/- 17.89%)**
	S_{0+}	**49.59% (+/- 20.69)%**	24.74% (+/- 15.45%)	*	36.18% (+/- 23.95%)
	S_{++}	**49.24% (+/- 13.92%)**	40.24% (+/- 10.92%)	23.75% (+/- 13.92%)	*

Table 7.2: Average difference between probabilities given to correct classes and probabilities given to wrong predicted classes.

Inspired by this conjecture, we conducted the methodology proposed by Martins et al. [2021] to identify *hard instances* in polarity classification tasks, which is able to identify texts with *discrepant* and *neutral* polarities. To do that, we selected 24 reviews for each of the scenarios that had the highest average probability error, which are: (i) instances of class S_{--}, but predicted as S_{++}; (ii) instances of class S_{0-}, but predicted as S_{++}; (iii) instances of class S_{0+}, but predicted as S_{--} and (iv) instances of class S_{++}, but predicted as S_{--}. Note that these are also the most distant classes from each other. The 24 instances chosen for each scenario are composed of the 12 instances with the highest error and 12 instances with the lowest error. In total, 96 instances were chosen.

A researcher with high peer review experience was asked to read the reviews, to score them (playing the role of the classifier) and indicate if she/he felt in doubt when rating them. According to Martins et al. [2021], reviews in which the annotator is in doubt about its polarity are *neutral hardest instances*. On the other hand, reviews for which the annotator is certain about its polarity and makes a mistake are *discrepant hardest instances*.

Human Prediction - All Reviews					
		Predicted Class			
	Review Class	S_{--}	S_{0-}	S_{0+}	S_{++}
Correct Class	S_{--}	**15**	4	4	1
	S_{0-}	5	**11**	3	5
	S_{0+}	7	7	**8**	5
	S_{++}	2	4	6	**12**
Accuracy	**47.91%**				

Table 7.3: Confusion matrix with all instances scored by the human.

Human Prediction Reviews with Highest Error					
		Predicted Class			
	Review Class	S_{--}	S_{0-}	S_{0+}	S_{++}
Correct Class	S_{--}	**6**	1	4	1
	S_{0-}	0	**7**	1	4
	S_{0+}	4	3	**5**	0
	S_{++}	2	2	**4**	**4**
Accuracy	**45.83%**				

Table 7.4: Confusion matrix with instances scored by the human, considering only the instances with the highest errors.

Human Prediction Reviews with Lowest Error					
		Predicted Class			
	Review Class	S_{--}	S_{0-}	S_{0+}	S_{++}
Correct Class	S_{--}	**9**	3	0	0
	S_{0-}	**5**	4	2	1
	S_{0+}	3	**4**	3	2
	S_{++}	0	2	2	**8**
Accuracy	**50%**				

Table 7.5: Confusion matrix with instances scored by the human, considering only the instances with the lowest errors.

Observe in Table 7.3 that the human classifier had an overall accuracy of 47% in instances where the model failed (0% accuracy). Also note that the human performed worse (Accuracy: 45.83%) on instances where the machine classifier error was higher (Table 7.4), when compared to the performance (Accuracy: 50%) on instances where the machine classifier error was lower (Table 7.5), indicating that, in fact, this group of instances may have more misleading texts.

Human Prediction Reviews without Doubt					
		Predicted Class			
	Review Class	S_{--}	S_{0-}	S_{0+}	S_{++}
Correct Class	S_{--}	**15**	2	1	1
	S_{0-}	5	**7**	0	5
	S_{0+}	**7**	2	**7**	2
	S_{++}	2	2	5	**12**
Accuracy	**54.66%**				

Table 7.6: Confusion matrix with instances scored by the human, considering only the instances that the human had no doubt.

Considering the two types of *hardest instances* proposed by Martins et al. [2021], for 21 (22%) instances the annotator was not certain about their polarities, that is, 22% of the instances in this sample are *neutral hard instances*. The accuracy for these instances was only 23% (Table 7.7), significantly lower than for the instances in which the annotator did not have any doubt: 54.66% (Table 7.6).

Human Prediction Reviews with Doubt					
		Predicted Class			
	Review Class	S_{--}	S_{0-}	S_{0+}	S_{++}
Correct Class	S_{--}	0	2	**3**	0
	S_{0-}	0	**4**	3	0
	S_{0+}	0	**5**	1	0
	S_{++}	0	**2**	1	0
Accuracy	**23.80%**				

Table 7.7: Confusion matrix with instances scored by the human, considering only the instances that the human had doubt.

Moreover, for 34 (35.41%) instances the annotator was certain about the polarity of the text, a mistake was made. These are the *discrepant hard instances*, which had a significance presence in our sample. For this instances, **both** the machine classifier and the human annotator made a mistake, and **both** were confident about their predictions.

7.2 Paper Decision Prediction (*PDP*)

Each cell of Table 7.8 shows the average confusion matrix for the *review score prediction* considering each of the *train* x *test* scenario, after a 5-fold cross validation.

BERT - Correct Class x Predicted Class (*PDP*)					
		Test			
	Scenario	Easy	Hard	Hardest	Hard + Hardest
Train	Easy	♡ 225.2; ♠ 10.2 ♣ 35.4; ♢ 29.4	♡ 182.4; ♠ 13.8 ♣ 60.8; ♢ 39.4	♡ 91.6; ♠ 11.4 ♣ 70.6; ♢ 36.8	♡ 268.6; ♠ 30.6 ♣ 125; ♢ 82.6
	Easy + Hard	♡ 222.2; ♠ 13.2 ♣ 26.6; ♢ 38.2	♡ 117.4; ♠ 18.8 ♣ 52.8; ♢ 47.4	♡ 91; ♠ 10.2 ♣ 60.8; ♢ 46.6	♡ 266.8; ♠ 32.4 ♣ 115.4; ♢ 92.2
	Easy + Hard + Hardest	♡ 217; ♠ 18.4 ♣ 24.2; ♢ 40.6	♡ 165.6; ♠ 30.6 ♣ 40.6; ♢ 59.6	♡ 83.2; ♠ 19.8 ♣ 52.2; ♢ 55.2	♡ 244.4; ♠ 54.8 ♣ 82.4; ♢ 125.2

Table 7.8: ♡: Rejected Papers, predicted as Rejected Papers; ♠: Rejected Papers, predicted as Accepted Papers; ♣: Accepted Papers, predicted as Rejected Papers; ♢: Accepted Papers, predicted as Accepted Papers. Confusion matrix of *PDP* task, with the average number of instances.

Taking the case Easy (*train*) x Easy (*test*) as an example, the cell is a confusion matrix which indicates that in this scenario: (♡) 222.2 papers predicted as rejected, were in fact rejected; (♠) 10.2 papers predicted as accepted, were actually rejected; (♣) 35.4 papers predicted as rejected, were actually accepted and (♢) 29.4 papers predicted as accepted, were in fact accepted. Therefore, the main diagonal (♡ and ♢) is composed of the cases where the classifier made a correct prediction, and the secondary diagonal (♠ and ♣) is composed of the cases where the classifier made an incorrect prediction.

That said, observe that *BERT* is more likely to make an incorrect classification of instances that indicate acceptance of a paper in a conference. This is expected, as the dataset is slightly imbalanced towards the rejection class. However, when *BERT* is trained

with *hardest instances*, the classifier precision on instances that were accepted increase significantly, it was able to predict acceptance decisions more correctly than incorrectly.

Table 7.9 describes the average magnitude of the error when the classifier makes a mistake in the *PDP* task. Observe that the magnitude of the error is considerably high in all scenarios, with the smallest difference is approximately 45%.

BERT - Classifier Error - Correct Class x Predicted Class (*PDP*)					
		Test			
	Scenario	Easy	Hard	Hardest	Hard + Hardest
Train	Easy	♠ 65.12%; ♣ 84.02%	♠ 57.85%; ♣ 77.54%	♠ 54.74%; ♣ 73.17%	♠ 53.58%; ♣ 72.40%
	Easy + Hard	♠ 58.48%; ♣ 78.41%	♠ 56.80%; ♣ 71.43%	♠ 45.47%; ♣ 61.40%	♠ 51.79%; ♣ 64.70%
	Easy + Hard + Hardest	♠ 55.25%; ♣ 70.62%	♠ 49.86%; ♣ 63.66%	♠ 52.63%; ♣ 61.60%	♠ 47.18%; ♣ 58.35%

Table 7.9: ♠: Rejected Papers, predicted as Accepted Papers; ♣: Accepted Papers, predicted as Rejected Papers. Average distance between probabilities given to correct classes and probabilities given to wrong predicted classes.

However, it is interesting to note that as more difficult instances are added in training or testing, there is a tendency for this error to decrease, indicating that the presence of *hard instances* can help to guide the perception of the classifier, although the absolute amount of incorrectly predicted instances is still large (Table 7.8).

In summary, the results shown in this section indicate that when the classifiers make mistakes, the distance between the incorrectly predicted class and the correct class is large, reinforcing that we are still far from having an autonomous peer review system. However, as shown in Section 7.1, humans have also difficulties to correctly classify the scores of reviews. Thus, for completely automatic peer review system to be deployed, maybe the entire process of writing a review should be rethought and redesigned to a more structured an unambiguous format, where the pros and cons of each paper are clearly stated, both semantically and in terms of their importance for the reviewer.

Chapter 8

Conclusion

8.1 Overview

In this work, we evaluate the performance of the state-of-the-art models in two tasks: *review score prediction* and *paper decision prediction*. In addition, we evaluated how they behave when exposed to more difficult instances (*hard instances*). The results of the experiments showed how important it is to pay attention to this type of instance if we want to have automated systems to help reviewers, editors and the scientific community in general. We believe that our findings can help our community to quantify how far we are to fully automated peer review systems, in which the reviewers only write their reviews and the system makes the rest of the decisions.

First, we investigated whether the performance of review score classifiers (*RSP*) is significantly impacted by review states. Next, we analyzed if the presence of reviews from different states also impacts the final decision regarding the paper (*PDP*). Our experiments revealed that, in general, *hard instances* of this nature do not significantly impact the performance of classifiers when added to training. This behavior is observed in both the *RSP* task and the *PDP* task. However, it is evident that the models have more difficulties in predicting the score and final decision of papers when the text of the reviews underwent modifications after the rebuttal phase. In the worst case, the difference was 12% in the *RSP* task and 23.31% in the *PDP* task.

We also investigated the impact of having a single borderline score (instead of the traditional *weak reject* and *weak accept* scores) in the review process. The idea is that this aggregation can make the task of reviewers easier. The results revealed that, as expected, borderline scores are the most difficult instances to be classified. The most difficult scenario for classifiers is to differentiate borderline acceptance score from borderline rejection score, indicating that the line between these two scores is quite small. The results also showed that transforming the two borderline classes into one, in general, does not significantly alter the classifier performance. This suggests that this borderline score division might not necessary, which can make the reviewers' task easier.

In order to present quantitative results regarding how far we are from having a system capable of automatically accepting or rejecting an paper, we investigated the confidence of the classifiers when they make mistakes. To do that, we computed the difference between the probabilities yielded by the classifier for the incorrectly predicted class and the correct class. The results revealed that when the classifier makes mistakes, this difference is usually high, indicating that the classifier makes mistakes with a high confidence. More specifically, we noticed that $\approx 50\%$ of the instances in the *RSP* task and $\approx 23\%$ of the instances in the *PDP* task were incorrectly classified with a high confidence. This indicates, therefore, that we are still far from a system capable of scoring a paper from its review and from automatically accepting or rejecting it. However, as shown in Section 7.1, the difficulties faced by machine classifiers are also faced by humans. This indicates that for the deployment of an automatic peer review system, it may be necessary to rethink the entire review writing process, so that the positive and negative characteristics of the papers are clearly informed, both semantically and in terms of their importance for the reviewer, in a way that the reviewer's position is clearly identified.

8.2 Future Work

We believe that the main direction for future work is the development of classification models that can classify more efficiently the hard instances presented in this work. Furthermore, it would be interesting that data from other conferences are analyzed and used to train existing or new models, in order to assess the particularities of each of these conferences, as well as the behavior of reviewers from other scientific communities.

As showed by Chakraborty et al. [2020], there is a strong correlation between the scores received by a paper and its final decision, which is why we use the report written by the reviewer, which should express the score, to train the model to predict the final decision. Nonetheless, we recognize that other information in the peer review process is considered in the paper's final decision, such as the reviewer's confidence, the author's answers in rebuttal phase, and the meeting with the reviewers and meta reviewers. We believe that in future works this information could be used in the training of a peer review automatic system.

We also recognize that a system fully responsible for predicting the score and the final decision of a paper, even considering the comments written by the reviewer, raises ethical questions that should be addressed. We understand that it requires a long and careful discussion, and this could be made in future works. Besides that, the comparison between the models and human performance in the tasks of predicting the score and the

final decision (as shown in Chapter 7 for the *review score prediction* task) could be more explored and use other experienced reviewers in the experiment.

Finally, we understand that over time, new models will be created in order to help reviewers and meta-reviewers in the tasks analyzed in this work, so it is essential that new works are developed with the objective of evaluating the performance of these new models and measuring how far we will be of an autonomous peer review classification system.

Bibliography

Jacalyn Kelly, Tara Sadeghieh, and Khosrow Adeli. Peer review in scientific publications: benefits, critiques, & a survival guide. *Ejifcc*, 25(3):227, 2014.

Publons. Global state of peer review. Technical report, Clarivate Analytics, 2018. URL https://doi.org/10.14322/publons.GSPR2018.

Karen White. Publications output: Us trends and international comparisons. science & engineering indicators 2020. nsb-2020-6. *National Science Foundation*, 2019.

Michail Kovanis, Raphaël Porcher, Philippe Ravaud, and Ludovic Trinquart. The global burden of journal peer review in the biomedical literature: Strong imbalance in the collective enterprise. *PloS one*, 11(11):e0166387, 2016.

Souvic Chakraborty, Pawan Goyal, and Animesh Mukherjee. Aspect-based sentiment analysis of scientific reviews. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, JCDL '20, page 207–216, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450375856. doi: 10.1145/3383583.3398541. URL https://doi.org/10.1145/3383583.3398541.

Richard Smith. Peer review: A flawed process at the heart of science and journals. *Journal of the Royal Society of Medicine*, 99(4):178–182, 2006. doi: 10.1177/014107680609900414. URL https://doi.org/10.1177/014107680609900414. PMID: 16574968.

Yang Gao, Steffen Eger, Ilia Kuznetsov, Iryna Gurevych, and Yusuke Miyao. Does my rebuttal matter? insights from a major NLP conference. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1274–1290, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1129. URL https://aclanthology.org/N19-1129.

Tirthankar Ghosal, Rajeev Verma, Asif Ekbal, and Pushpak Bhattacharyya. DeepSentiPeer: Harnessing sentiment in review texts to recommend peer review decisions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1120–1130, Florence, Italy, July 2019a. Association for Computational Linguistics. doi: 10.18653/v1/P19-1106. URL https://aclanthology.org/P19-1106.

Zhongfen Deng, Hao Peng, Congying Xia, Jianxin Li, Lifang He, and Philip Yu. Hierarchical bi-directional self-attention networks for paper review rating recommendation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6302–6314, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.555. URL https://aclanthology.org/2020.coling-main.555.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630*, 2015. URL https://arxiv.org/pdf/1511.08630.pdf.

Xingyou Wang, Weijie Jiang, and Zhiyong Luo. Combination of convolutional and recurrent neural network for sentiment analysis of short texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2428–2437, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee. URL https://aclanthology.org/C16-1229.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf.

Gianni Brauwers and Flavius Frasincar. A survey on aspect-based sentiment classification. *ACM Comput. Surv.*, nov 2021. ISSN 0360-0300. doi: 10.1145/3503044. URL https://doi.org/10.1145/3503044.

DataReportal. Digital 2020 global digital overview. Technical report, Kepios Pte. Ltd., 2020. URL https://datareportal.com/reports/digital-2021-global-digital-overview.

Patricio Cerda, Gaël Varoquaux, and Balázs Kégl. Similarity encoding for learning with dirty categorical variables. *Machine Learning*, 107(8):1477–1494, 2018.

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. Text classification algorithms: A survey. *Information*, 10 (4):150, 2019.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, aug 1988. ISSN 0306-4573. doi: 10.1016/0306-4573(88)90021-0. URL https://doi.org/10.1016/0306-4573(88)90021-0.

Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003.

Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

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

Femi Joseph and N Ramakrishnan. Text categorization using improved k nearest neighbor algorithm. *Int J Trends Eng Technol*, 4:65–68, 2015.

Ammar Ismael Kadhim. Survey on supervised machine learning techniques for automatic text classification. *Artificial Intelligence Review*, 52(1):273–292, 2019.

Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saied Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv preprint arXiv:1707.02919*, 2017.

Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001.

William S Noble. What is a support vector machine? *Nature biotechnology*, 24:1565–1567, 2006.

James M Keller, Michael R Gray, and James A Givens. A fuzzy k-nearest neighbor algorithm. *IEEE transactions on systems, man, and cybernetics*, pages 580–585, 1985.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Azzurra Ragone, Katsiaryna Mirylenka, Fabio Casati, and Maurizio Marchese. On peer review in computer science: Analysis of its effectiveness and suggestions for improvement. *Scientometrics*, 97(2):317–356, November 2013. ISSN 0138-9130. doi: 10.1007/s11192-013-1002-z. URL https://doi.org/10.1007/s11192-013-1002-z.

Andrew Tomkins, Min Zhang, and William D. Heavlin. Reviewer bias in single- versus double-blind peer review. *Proceedings of the National Academy of Sciences of the United States of America*, 114(48):12708–12713, 2017. doi: 10.1073/pnas.1707323114. URL https://doi.org/10.1073/pnas.1707323114. PubMed: 29138317.

Pali UK De Silva and Candace K Vance. Preserving the quality of scientific research: peer review of research articles. *Scientific Scholarly Communication*, 99(4): 73–99, 2017. doi: 10.1007/978-3-319-50627-2_6. URL https://doi.org/10.1007/978-3-319-50627-2_6.

John Langford and Mark Guzdial. The arbitrariness of reviews, and advice for school administrators. *Commun. ACM*, 58(4):12–13, March 2015. ISSN 0001-0782. doi: 10.1145/2732417. URL https://doi.org/10.1145/2732417.

Laurent Charlin and Richard Zemel. The toronto paper matching system: An automated paper-reviewer assignment system. *ICML Workshop on Peer Reviewing and Publishing Models (PEER)*, 28, 2013. URL https://mila.quebec/wp-content/uploads/2016/03/tpms.pdf.

Omer Anjum, Hongyu Gong, Suma Bhat, Wen-Mei Hwu, and JinJun Xiong. PaRe: A paper-reviewer matching approach using a common topic space. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 518–528, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1049. URL https://aclanthology.org/D19-1049.

Tirthankar Ghosal, Debomit Dey, Avik Dutta, Asif Ekbal, Sriparna Saha, and Pushpak Bhattacharyya. A multiview clustering approach to identify out-of-scope submissions in peer review. In *2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 392–393, 2019b. doi: 10.1109/JCDL.2019.00086.

Sandeep Kumar, Tirthankar Ghosal, Prabhat Kumar Bharti, and Asif Ekbal. Sharing is caring! joint multitask learning helps aspect-category extraction and sentiment de-

tection in scientific peer reviews. In *2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 270–273, 2021a. doi: 10.1109/JCDL52503.2021.00081.

Shruti Singh, Mayank Singh, and Pawan Goyal. Compare: A taxonomy and dataset of comparison discussions in peer reviews. In *2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 238–241, 2021. doi: 10.1109/JCDL52503.2021.00068.

Chaitanya Bhatia, Tribikram Pradhan, and Sukomal Pal. *MetaGen: An Academic Meta-Review Generation System*, page 1653–1656. Association for Computing Machinery, New York, NY, USA, 2020. ISBN 9781450380164. URL https://doi.org/10.1145/3397271.3401190.

Asheesh Kumar, Tirthankar Ghosal, and Asif Ekbal. A deep neural architecture for decision-aware meta-review generation. In *2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 222–225, 2021b. doi: 10.1109/JCDL52503.2021.00064.

Alberto Bartoli, Andrea De Lorenzo, Eric Medvet, and Fabiano Tarlao. Your paper has been accepted, rejected, or whatever: Automatic generation of scientific paper reviews. In Francesco Buccafurri, Andreas Holzinger, Peter Kieseberg, A Min Tjoa, and Edgar Weippl, editors, *Availability, Reliability, and Security in Information Systems*, pages 19–28, Cham, 2016. Springer International Publishing. ISBN 978-3-319-45507-5.

Bela Gipp, Corinna Breitinger, Norman Meuschke, and Joeran Beel. Cryptsubmit: introducing securely timestamped manuscript submission and peer review feedback using the blockchain. In *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pages 1–4, Toronto, ON, Canada, 2017. IEEE, IEEE.

Nicholas Weber and Sebastian Karcher. *Seeking Justification: How Expert Reviewers Validate Empirical Claims with Data Annotations*, page 227–234. Association for Computing Machinery, New York, NY, USA, 2020. ISBN 9781450375856. URL https://doi.org/10.1145/3383583.3398537.

Kimitaka Tsutsumi, Kazutaka Shimada, and Tsutomu Endo. Movie review classification based on a multiple classifier. In *Proceedings of the 21st Pacific Asia Conference on Language, Information and Computation*, pages 481–488, Seoul National University, Seoul, Korea, November 2007. The Korean Society for Language and Information (KSLI). doi: http://hdl.handle.net/2065/29106. URL https://aclanthology.org/Y07-1050.

Shuvashish Paul Sagar, Khondokar Oliullah, Kazi Sohan, and Md Fazlul Karim Patwary. Prcmla: Product review classification using machine learning algorithms. In *Proceedings of International Conference on Trends in Computational and Cognitive Engineering*, pages 65–75. Springer, 2021. doi: 10.1007/978-981-33-4673-4_6. URL https://doi.org/10.1007/978-981-33-4673-4_6.

Mohammad Al-Smadi, Omar Qawasmeh, Mahmoud Al-Ayyoub, Yaser Jararweh, and Brij Gupta. Deep recurrent neural network vs. support vector machine for aspect-based sentiment analysis of arabic hotels' reviews. *Journal of computational science*, 27:386–393, 2018. doi: 10.1016/j.jocs.2017.11.006. URL https://doi.org/10.1016/j.jocs.2017.11.006.

Gayatree Ganu, Noemie Elhadad, and Amélie Marian. Beyond the stars: Improving rating predictions using review text content. In *12th International Workshop on the Web and Databases, WebDB 2009, Providence, Rhode Island, USA, June 28, 2009*, 2009. URL http://webdb09.cse.buffalo.edu/papers/Paper9/WebDB.pdf.

Ke Wang and Xiaojun Wan. Sentiment analysis of peer review texts for scholarly papers. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 175–184, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356572. doi: 10.1145/3209978.3210056. URL https://doi.org/10.1145/3209978.3210056.

David Soergel, Adam Saunders, and Andrew Mccallum. Open scholarship and peer review: a time for experimentation. *ICML Workshop on Peer Reviewing and Publishing Models (PEER)*, 28, 2013. URL https://openreview.net/pdf?id=xf0zSBd2iufMg.

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. A dataset of peer reviews (PeerRead): Collection, insights and NLP applications. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1647–1661, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1149. URL https://aclanthology.org/N18-1149.

Xinyu Hua, Mitko Nikolov, Nikhil Badugu, and Lu Wang. Argument mining for understanding peer reviews. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2131–2137, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1219. URL https://aclanthology.org/N19-1219.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657, 2015.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal 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), pages 1681–1691, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1162. URL https://aclanthology.org/P15-1162.

Michael R Smith, Tony Martinez, and Christophe Giraud-Carrier. An instance level analysis of data complexity. *Machine learning*, 95(2):225–256, 2014.

Beata Beigman Klebanov and Eyal Beigman. Squibs: From annotator agreement to noise models. *Computational Linguistics*, 35(4):495–503, 2009. doi: 10.1162/coli.2009.35.4.35402. URL https://aclanthology.org/J09-4005.

Beata Beigman Klebanov and Eyal Beigman. Difficult cases: From data to learning, and back. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 390–396, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2064. URL https://aclanthology.org/P14-2064.

Karen Martins, Pedro O.S Vaz-de Melo, and Rodrygo Santos. Why do document-level polarity classifiers fail? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1782–1794, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.143. URL https://aclanthology.org/2021.naacl-main.143.

Appendix A

Vector Fill Approaches

In addition to completing with vectors filled with the average of the probabilities, we also tested the performance of the MLP when using zero-filled vectors and 0.5-filled vectors. Table A.1 presents the results achieved using these approaches.

Overall Results			
	Review Score Prediction (RSP)		**Paper Decision Prediction (PDP)**
Model	**Input: Text**	**Vector Fill**	**Input: RSP Output**
LSTM	24.81% (+/- 1.28%)	**Average**	66.25% (+/- 0.06%)
		0.5	
		0.0	
RNN-GRU	25.03% (+/- 0.67%)	**Average**	66.25% (+/- 0.06%)
		0.5	
		0.0	
XLNet	44.26% (+/- 0.94%)	**Average**	73.88% (+/- 0.92%)
		0.5	74.20% (+/- 1.27%)
		0.0	73.53% (+/- 1.55%)
RoBERTa	45.93% (+/- 1.08%)	**Average**	74.85% (+/- 0.65%)
		0.5	74.77% (+/- 1.44%)
		0.0	74.92% (+/- 1.61%)
BERT	**49.20% (+/- 1.00%)**	**Average**	**76.88% (+/- 0.99%)**
		0.5	76.31% (+/- 0.71%)
		0.0	76.46% (+/- 0.91%)

Table A.1: Accuracy of models in *RSP* and *PDP* tasks using other approaches to fill the vector.

Appendix B

Training Information

B.1 Model Repository

The code for all classifiers used in this work is available on GitHub. As already explained, modifications were necessary so that all models were evaluated according to the same parameters. Models: *DeepSentiPeer*, *HabNet*, *C-LSTM*, *CNN-GRU*, *XLNet*, *RoBERTa* and *BERT*.

B.2 Model Training Parameters

The experiments were performed on an Intel Core i7-9700KF CPU 3.40 GHz and NVIDIA GeForce RTX 2080 Ti 11GB machine.

The *DeepSentiPeer* and *HabNet* models were trained following the specifications described in their work repositories. The *C-LSTM* model was trained with the following configuration: batch size of 32, 100 filters with size 4 in the convolutional layer, and 200 as memory dimension for LSTM. The *CNN-GRU* model was trained with the following configuration: 100 filters with size 5, and 3 as pool size for both CNNs. In the GRU, dimensionality of 150 and batch size of 32. The *XLNet* model was trained with the following configuration: batch size of 8, max sequence length of 256 and pre-trained model xlnet-base. The *RoBERTa* model was trained with the following configuration: batch size of 16, max sequence length of 256 and pre-trained model roberta-base. The *BERT* model was trained with the following configuration: batch size of 16 and max sequence length of 512, and pre-trained model bert-base.