Discourse Segmentation of Judicial Decisions for Fact Extraction

Andrés Lou

A Thesis
in
Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Master of Computer Science at
Concordia University
Montréal, Québec, Canada

April 2021

© Andrés Lou, 2021

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Andrés Lou**

Entitled: **Discourse Segmentation of Judicial Decisions for Fact Extraction**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

______________________________ Chair
Dr. Peter Rigby

______________________________ Examiner
Dr. Leila Kosseim

______________________________ Examiner
Dr. Olga Ormandjieva

______________________________ Thesis Supervisor
Dr. Leila Kosseim

Approved by ______________________________
Dr. Hovhannes Harutyunyan, Graduate Program Director

July 1, 2019 ______________________________
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Discourse Segmentation of Judicial Decisions for Fact Extraction

Andrés Lou

In order to justify rulings, legal documents need to present facts as well as an analysis built on these. However, identifying what a *Fact* is and what a *non-Fact* is, and how they relate in order to bring together a legal document is often not an easy task. Because of a number of reasons, such as the domain-specific definition of the term *Fact*, or the technical vocabulary that permeates the documents in which these are to be found, extracting the case-related facts from a legal document is usually time-consuming and expensive.

In this thesis, we present two methods to automatically extract case-relevant facts in French-language legal documents pertaining to tenant-landlord disputes. We base our approaches on the assumption that the text of a decision will follow the structural convention of first stating the facts and then performing an analysis based on them. This assumption is itself based on the widespread application of the IRAC legal document writing model and its many variants (Beazley 2018). Given a legal document, we perform text segmentation to extract the parts that contain the case-relevant *Facts* using two different approaches based on neural methods commonly used in Natural Language Processing and a novel heuristic method based on the density of *Facts* in a segment of the text.

Our two approaches are based on the representation of legal texts as binary strings, where contiguous subsequences of 1's represent sentences containing *Facts* and, conversely, contiguous subsequences of 0's correspond to sentences containing *non-Facts*. The first approach consists of classifying each sentence in the document as either *Facts* or *non-Facts* using an ensemble model of independent word embeddings (Mikolov et al. 2013), GRU networks (Cho et al. 2014) and Convolutional Neural Networks (Kim 2014). The second approach consists of a contextual classification of sentences as either class using recurrent architectures to create the binary string representation of the document; we experiment using LSTM networks (Hochreiter and Schmidhuber 1997), GRU networks, and Attention Encoder-Decoder models (Bahdanau, Cho, and Bengio 2014). The segmentation is carried out by introducing the concept of *purity*, a measure of the density of facts in a given subsequence in the binary string; the facts are extracted by maximising both the length and the purity of the substring containing the facts.

Extrinsic evaluations of both approaches show that the second approach outperforms the first by producing the greater number of documents whose segmentation point is predicted within a single sentence of difference from the one indicated by the gold standard. Nevertheless, a significant percentage of segmentation points are underestimated by being predicted more than four sentences away from the point determined by the gold standard.

Acknowledgments

I would like to thank my tireless supervisor, professor Leila Kosseim, for her continued guidance. The value of her advice and the moral support that she provided cannot be overstated. She is one of the reasons my Master's degree was an overall great experience.

I would like to thank my fellow labmates at the Computational Linguistics at Concordia Lab, with whom my knowledge of both our field and the world in general was greatly expanded. In particular, I must mention my colleague and friend Farhood Farahnak, whose expertise in the field is only matched by his talent for teaching and his willingness to share his knowledge with whomever needs a helping hand.

Last, but by no means least, I would like to thank my wife, Martha, for her relentless support in the face of our shared adversity. A graduate student herself, she is and endless source of understanding and loving support, without whom none of this would have been possible.

A special mention goes to my dearest kitty-cat, Ludmila.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Understanding the rationale behind a particular ruling made by a judge is a complex task that requires formal legal training in the relevant case law. Nevertheless, the rulings are still made using traditional methods of human discourse and reasoning, including default logic (Walker, Lopez, et al. 2015), deontic logic (Dragoni et al. 2015), and rhetoric (Wald 1995). In particular, knowing all the relevant facts surrounding a case is of the utmost importance to understand the outcome of a ruling, as they are necessary to arrive at the best-possible decision, since facts are what give way to what is usually called the *Best Evidence* (Stein 2005; Nance 1987).

Within a ruling, however, determining what constitutes a fact, as opposed to other types of content that motivate and illustrate a judge's decision, is also a matter that requires formal training. Indeed, a variety of linguistic factors, such as specialised terminology, textual structure, linguistic register, as well as domain knowledge, make the ruling stray from more general-domain texts. As such, fact extraction from legal texts is time consuming, expensive, and requires legal expertise. Additionally, as Westermann et al. (2019) have shown, even amongst trained experts, inter-annotator agreement tends to be low. For example, in the task of labelling a corpus of rulings with a pre-established set of labels on the ruling's subject matter, Westermann et al. (2019) reported low inter-annotator agreement and suggested that its cause might be the general vagueness and lack of explicit reasoning in the texts.

1.1.1 What is a *Fact*?

There are several definitions of what constitutes a *fact*, many of which are epistemological in nature. Here, we constrain the working definition to the realm of Law. According to the Merriam-Webster dictionary's legal definition (Merriam-Webster 2020), a **fact** is:

> "Any of the circumstances of a case that exist or are alleged to exist in reality : a thing whose actual occurrence or existence is to be determined by the evidence presented at trial."

According to ALM's Legal Dictionary (Law.com 2020), a fact is:

> "An actual thing or happening, which must be proved at trial by presentation of evidence and which is evaluated by the finder of fact (a jury in a jury trial, or by the judge if he/she sits without a jury)."

Besides these general definitions, there are different kinds of facts, depending on the application and relevance of the legal context in which it plays a role (eg Merriam-Webster's Legal Dictionary (Merriam-Webster 2020) presents definitions for the terms *Adjudicative Fact, Collateral Fact, Constitutional Fact*, etc). However, this thesis will deal with the general concept of a fact, as it stands in contrast with an *analysis* and any other reason for acting that might guide the decision made by a judge.

Facts alone do not determine the outcome of a decision; they must be accompanied by an adequate *classification* because not all facts are relevant to the issue at hand (Hage, Waltermann, and Akkermans 2017, page 23). Nevertheless, facts are the basis for any kind of judicial application of the rules. Consider the example taken from (Hage, Waltermann, and Akkermans 2017):

> **Example 1.** Pierre, 18 years old, visits the house of his neighbors. He is distracted by the daughter of the house, does not look where he is walking, stumbles over the carpet, and falls against an antique Chinese vase, which breaks as a consequence. The value of the vase was €3,000. Must Pierre (or his insurance) pay his neighbor €3,000?

To answer the question from Example 1, we can construe the facts as follows:

- Pierre is 18 years old.
- He visited the neighbours' house.
- He became distracted because of the neighbours' daughter.
- He stumbled over the carpet and fell against an antique Chinese vase.
- The vase broke.
- The vase was valued at €3,000.

Using these facts, Hage, Waltermann, and Akkermans (2017) determine that, if the rule in question is:

> "IF somebody acted wrongfully toward another person, and if they thereby caused damage to this other person, THEN they must compensate this damage."

then Pierre has “acted wrongfully toward his neighbors and thereby he caused damage to his neighbors”, and therefore the logical conclusion is that “Pierre must compensate the damage to his neighbors.”

Even using a relatively straightforward definition, however, determining whether a given statement, assuming complete truthfulness, actually contains a fact is not always simple, especially since, as with any specialised field, a technical document will follow a written convention that may not be as straightforward as the one shown in Example 1. Consider the following, real-life, example taken from the corpus of the *Régie du logement du Québec* (see Section 3.1)

> **Example 2.** "The landlord demands the termination of the lease and the eviction of the tenant, the recovery of the rent (1,080 $) as well as the rent owed at the time of the audience, plus the provisional execution of the decision despite de outcome. This is concerning a lease from December 1 2016 to June 30 2018 at a monthly rent of 545 $, to be payed the first of each month. The proof shows that the tenant owes 1,620 $, this being the rent corresponding to the months of February, March, and April 2017, plus 9 $ corresponding the costs of notification as stipulated by the Rules. The tenant is more than three months late in their rent payments, thus the termination of the lease is justified by applying Article 1,971 of the Civil Code of Québec. The lease is however not terminated if the owed rent, the interests, and the costs, are payed before the decision, conforming to the dispositions laid down by Article 1,883 of the Civil Code of Québec. The injury caused to the landlord justifies the provisional execution of the decision, as foreseen by Article 82.1 of the Law on the *Régie du logement*. Can the tenant's lease be terminated? Can it be terminated immediately, as the landlord requests? "

Because of the language, the complexity of the case, and the length of the text, construing the facts of the case is not as straightforward as in Example 1. Moreover, this is a relatively short and simple, real-life case; longer cases can span thousands of words and describe much more complex situations.

1.1.2 The sentence

The basic unit in the text that contains a fact is the sentence; this can be observed in both Examples 1 and 2. In this context, a sentence is understood to be a self-contained grammatical unit usually delimited by a period, though sometimes other punctuation markers, such as semicolons or dashes, might be used. While a sentence may contain more than one fact, it is not likely that a single fact spans multiple sentences. Thus, a part of the challenge of extracting facts from a case is to determine whether a given sentence contains one or several facts or not.

As mentioned previously, this is challenging for the non-expert, because of the complexities of style, vocabulary, domain knowledge, and real-world knowledge. As a real-world example of the challenge of labelling sentences as either Facts or non-Facts, consider the samples in Figure 1.

Figure 1 shows a segment of a real-like example taken from the corpus of the *Régie du logement* (see Section 3.1), where four sentences were labelled as either containing *Facts* or *non-Facts* by the author of the ruling (in this case, the judge). We can see the difficulty in classifying each sentence because of the use of technical words (eg *mandate*, *emanates*), domain knowledge (the citation of specific articles of law), and real-world knowledge (the fact that the judge is aware that the <ORG> entities are distinct).

1.2 IRAC and the base assumption

The IRAC standard model (“Issue-Rule-Application-Conclusion”) is a syllogistic formula for organising legal analysis (Beazley 2018). It is a standard of teaching in many first-year law introductory courses and it has many variants across the different publications and schools of thought (cf (Beazley 2018), (Felsenburg and Graham 2010) and (Flanigan

Original sentence	**English translation**	**Tag**
Comme le mandat fourni à l'audience par Madame <NAME> émane de <ORG>. qui n'est pas le véritable locateur, ce mandat n'est pas conforme à l'article 72 de la Loi sur la Régie du logement.	*As the mandate provided at the hearing by Ms. <NAME> emanates from <ORG>, who is not the real landlord, the mandate does not comply with section 72 of the Act respecting the Régie du logement.*	Fact
Rien ne prouve par ailleurs que <NAME> est employée de <ORG> et <ORG> puisque ces compagnies ne lui ont pas donné de mandat.	*There is also no evidence that <NAME> is an employee of <ORG> and <ORG> since these companies did not give him a mandate.*	Fact
Ceci étant dit, revenons à l'argumentation de Monsieur <NAME> et de Madame <NAME> voulant que toutes ces compagnies soient liées entre elles et puissent représenter l'autre sans plus de formalité.	*That said, let us return to the argument of <NAME> and <NAME> that all of these companies are linked and can represent the other without further formality.*	non-Fact
Cet argument ne tient pas; en effet, même si les compagnies sont dirigées par les mêmes personnes et que l'une soit l'actionnaire majoritaire de l'autre, il n'en demeure pas moins qu'il s'agit d'entités juridiques distinctes.	*This argument does not hold; even if the companies are managed by the same people and one is the majority shareholder of the other, the fact remains that they are separate legal entities.*	non-Fact

Figure 1: Example of sentence classification as either stating a case Fact or non-Fact. (English translations provided by the authors. For privacy reasons, proper names have been redacted.)

2019)). Although its use is more widespread in academic circles (Garner 2009), it constitutes a good starting point on the writing of legal documents by providing a theoretical framework with which to craft legal arguments. In its most basic form, the IRAC standard model follows the structure illustrated in Example 3:

Example 3.

Issue:	Is Lúthien immortal?
Rule:	Men are mortal, elves are immortal
Application:	Lúthien is an elf-maiden
Conclusion:	Lúthien is immortal

The legal document explaining the issue and resolution of this matter would follow the four stages of the IRAC framework. However, as previously mentioned, IRAC has many variants, which consist of adding other components in order to refine the relation between the rule (*Men are mortal, elves are immortal*) and the application (*Lúthien is an elf-maiden*). For example, if we introduce a new component, the *Explanation* (Beazley 2018, p. 94), into de framework, we can refine the original syllogism as follows:

Example 4.

Issue:	Is Lúthien immortal?
Rule:	Men are mortal, Elves are immortal
Explanation:	Elf-maidens are Elves
Application:	Lúthien is an Elf-maiden
Conclusion:	Lúthien is immortal

The *Rule* and the *Explanation* in both Examples 3 and 4 correspond to facts which, when applied, permits the drawing of a conclusion. However, the application and the associated explanation are not ontologically necessary to the other components: a rule may exist whether there is a particular application of it or not, just as Men would remain mortal and Elves would remain immortal even if there were no Elf-maiden named Lúthien from whose mortality a conclusion could be drawn. In the context of this thesis, the *Rules*, *Explanations* and *Applications* in Examples 3 and 4 correspond to the real-life facts found in each of the documents in our corpus (see Chapter 3), as illustrated in Example 5.

Example 5.

non-Fact: Is Lúthien immortal?

Fact: Men are mortal, Elves are immortal

Fact: Elf-maidens are Elves

Fact: Lúthien is an Elf-maiden

non-Fact: Lúthien is immortal

According to the IRAC framework and its variants, these should exist as semantically independent and distinguishable units of discourse, ie sentences. Moreover, if the framework, and its variants, are as widespread as the literature suggests, these should occupy the same place, relative to the body of text, in each legal document in our corpus, namely, before any sort of *Conclusion* is presented. Thus, Figure 2 presents the basic assumption of this thesis.

Assumption. *The text of a decision will follow the structural convention of first stating the facts and then performing an analysis based on them.*

Figure 2: The basic assumption of this thesis

The assumption that the text of a decision will follow the structural convention of first stating the facts and then performing an analysis based on them is intuitive, since a case analysis, that is, its explanation, application and conclusion, is built from its facts, which gives way to the best decisions. Hence, the sentences of such decisions should follow this discourse structure.

1.3 The JusticeBot project

The JusticeBot project (Westermann et al. 2019) is an initiative by the CyberJustice Lab at the Université de Montréal that aims to provide a gateway to law and jurisprudence for lay people through a chatbot where users can seek remedies to terminate their lease because of landlord-tenant disputes. As part of the extrinsic evaluation of the system as a whole, a *challenge task* was conceived whereby participants would be asked to build predictive systems that would be required to answer a number of questions related to the outcome of a given case. For example, given a document, or the relevant case facts of a documents, participants would have to predict whether the judge ruled in favour of the tenant, the amount of the judge order as compensation to either the plaintiff or the defendant, length of a sentence, etc. The challenge tasks are in this manner akin to the shared tasks of SensEval (Edmonds 2002) and SemEval (Strapparava and Mihalcea 2007). Prior to the deployment of the challenge, a system capable of extracting case-related facts in order to build a training corpus for the task was required.

1.4 Goal of this thesis

Starting from the assumption in Figure 2, this thesis proposes a set of approaches to identify and segment *Facts* in texts of judicial rulings.

Specifically, given the text of a judicial decision, using the principles of IRAC and its variants, and assuming the document follows the convention of listing all relevant case facts before the analysis and conclusion, this thesis seeks to extract the case relevant facts by means of neural approaches for text segmentation in the legal domain. We approach the problem using two different methodologies:

- By performing individual Binary Sentence Classification to determine which segment of the text contains the *Facts* and which contains the *non-Facts*, where the outcome of each sentence is independent of the outcome of all others (see Chapter 4).

- By performing the segmentation using a recurrent architecture to classify each sentence in the document as either *Facts* or *non-Facts*, where the outcome of each classification is conditioned by the context in which the sentence is found (see Chapter 5).

We evaluate both methodologies using intrinsic evaluation metrics (accuracy, precision, recall, and F_1) and an extrinsic metric consisting of counting the number of documents whose segmentation is off by a varying number of sentences.

1.5 Contributions

This thesis presents the following contributions:

- It introduces neural approach to a relevant task in the intersection of Natural Language Processing and Law: the segmentation of legal texts. We present two novel methodologies based on the principles Sentence Classification, Contextual and non-Contextual Embeddings applied to the task of extracting factual information from legal cases concerning tenant-landlord disputes.

- It builds on and contributes to the JusticeBot, a project developed by the CyberJustice Laboratory with the aim of simplifying access to legal information for the public. At the time of writing, JusticeBot is focused on facilitating the access to information concerning landlord-tenant disputes. In particular, this thesis provides a methodology by which the training and testing datasets of the challenge task can be built (see Section 1.3).

1.6 Thesis structure

This chapter has briefly described the nature and challenges of extracting case-relevant facts from judicial decisions, and the assumption on which we base the Text Segmentation task. The rest of this thesis is structured as follows: Chapter 2 explores the work done in the fields of Law and AI in terms of the application of Natural Language Processing, with special emphasis on neural methods; Chapter 3 describes the nature, acquisition, and processing of the corpus we use to train and evaluate the models on which our approaches are based; Chapter 4 describes in detail the principle, training, and evaluation of the first approach, the Sentence Classifier method; Chapter 5 describes in detail the principle, training, and evaluation of the second approach, the Recurrent Architecture; finally, Chapter 6 provides a summary of the thesis and proposes improvements and future work.

Chapter 2

Related work

This chapter will expose the work that has been done at the intersection of Law and Natural Language Processing (NLP) and present a brief introduction to the neural methods used in this thesis to implement the methodologies described in Chapters 4 and 5. Section 2.1 will describe the application of traditional NLP methods for a variety of tasks involving the legal domain. Section 2.2 will describe work done in Sentence Classification and Text Segmentation. Section 2.3 will present an overview of recent developments in neural methods for NLP: Section 2.3.1 will describe the neural models used in our proposed first approach, based on the context-independent binary classification of individual sentences, while Sections 2.3.1 and 2.3.3 will describe the models used for our second proposed approach, based on a recurrent architecture that allows for the content-dependent and context-dependent classification of sentences.

2.1 Natural Language Processing in legal environments

The use of Natural Language Processing (NLP) to mine judicial corpora is not new; however, very little work has used neural methods, as most of the cited literature uses rules or hand-crafted features to perform their stated tasks. Maat, Winkels, and Van Engers (2006) developed a parser that automatically extracts reference structures (which is loosely equivalent to the notion of named entities) in and between legal sources. A few years later, Maat and Winkels (2009) developed a model based on syntactic parsing that automatically classifies norms in legislation by recognising typical sentence structures. Dell'Orletta et al. (2012) proposed a shared task on dependency parsing of legal texts and domain adaptation where participants compared different parsing strategies utilising the DeSR parser (Attardi 2006) and a weighted directed graph-based model (Sagae and Tsujii 2010). Grabmair et al. (2015) demonstrated the feasibility of extracting argument-related semantic information and used it to improve document retrieval systems. Walker, Han, et al. (2017) introduced an annotated dataset based on propositional connectives and sentence roles in veterans' claims, wherein each document is annotated using a typology of propositional connectives and the frequency of the sentence types that led to adjudicatory decisions. Jaromír Savelka and Ashley (2017) used Conditional Random Fields (CRF) to extract sentential and non-sentential content from decisions of United States courts, and also extract functional and issue-specific segments (Jaromír Savelka and Ashley 2018), achieving near-human performance. Finally, Dragoni et al. (2015) performed rule-extraction from legal documents using a combination of a syntax-based system using linguistic features provided by WordNet (Miller 1995), and a logic-based system that extracts dependencies between the chunks of their dataset. Their work is closely related to our task; however, whereas Dragoni et al. (2015) base their model on syntactic and logical rulesets, we base our model on Recurrent and Convolutional Neural models (see Sections 2.3.1 and 2.3.1), introduce our own segmentation heuristic (see Section 4.3), and use independent and contextual word embeddings (Mikolov et al. 2013; Martin et al. 2019).

Of note is the work of Campbell et al. (2018), who developed *ProceZeus*, a chatbot intended to facilitate access to judicial proceedings involving rental housing issues in Québec.

ProceZeus is able to build an input case by asking a series of questions to the user and afterwards presenting the most similar cases to that of the user using k-Nearest Neighbours (Fix and Hodges 1952), predicting the potential monetary compensation using regression, and using Support Vector Machines (Cortes and Vapnik 1995) to predict the outcome of the case.

2.2 Text segmentation and text classification

Most sentence classification tasks in the literature are concerned with sentiment analysis. Our task, in contrast, is annotated following an arguably less intuitive scheme: whether a given sentence seems to state a case fact or if it represents a part of the judicial analysis (discussed in more detail in Chapter 3). Sentiment analysis itself is faced with the constant issue of inter-annotator disagreement (cf (Westermann et al. 2019), (Mozetič, Grčar, and Smailović 2016) and (Bermingham and Smeaton 2009)), and while sentiment analysis and opinion mining have been very popular for many years now (see (Pang, Lee, et al. 2008) for a detailed survey of the literature), classification tasks involving more abstract, or less polarised, categories are not nearly as common.

Text segmentation is a task that consists of dividing a document into meaningful syntactic or semantic units at the word or sentence levels. Topic modelling, in particular, has been a fundamental task of the field for some time now (Ponte and Croft 1997), and it is by far the most common application of text segmentation. As part of their Topic Detection and Tracking study, Allan et al. (2003) developed a model that segments a stream of text from speech transcriptions using Hidden Markov Model techniques. Similarly, Beeferman, Berger, and Lafferty (1999) introduced statistical methods in the form of featured-based models that predict text boundaries using the metric of *topicality*, a measure of the ratio of predictions between a long-range language model and a trigram model. In contrast to the domain-specific nature and use of lexical cohesion of these works, Utiyama and Isahara (2001) developed a domain-independent segmentation model intended to be used on text summarisation based solely on statistical language models, which was shown to match and even outperform previous work. Recent work based on stochastic methods mostly use models based on Latent Dirichlet Allocation, a statistical generative model where each item in a collection is modelled as an infinite mixture over an underlying set of topics (Blei, Ng, and Jordan 2003; Misra et al. 2009). In contrast to this previous work, our segmentation method is based on accounting for the varying density of factual content across the many possible segmentations that can be produced from a single text.

In order to better understand our proposed approaches, we will now briefly introduce the foundations and the principles of the neural methods we used to carry out the task of text segmentation.

2.3 Neural methods for NLP

Outside the frame of legal texts, semantic sentence classification has recently achieved new benchmarks thanks to the application of neural methods. The use of Recurrent Neural Networks (RNN), and their derivative recurrent architectures (in particular encoder-decoder models (Cho et al. 2014)), has steadily produced results that outperform traditional models in many NLP tasks such as Machine Translation, Question Answering and Text Summarisation (eg the Attention mechanism Bahdanau, Cho, and Bengio 2014 and the Sequence-to-Sequence model (Sutskever, Vinyals, and Le 2014), two of the most important architectural developments in RNNs). Sentence classification using Deep Learning (DL) was first proposed by Kim (2014), who used Convolutional Neural Networks (CNN) and showed that their models outperformed classical models on many standard datasets. Their work was quickly followed by the application of RNN architectures for similar tasks, including those of Lai et al. (2015), Zhang, Zhao, and LeCun (2015), and Zhou et al. (2016). A major breakthrough was achieved with the Transformer (Vaswani et al. 2017) (see Section 2.3.3), whose non-recurrent, Multi-Head Attention architecture allowed for richer language model representations that capture many more linguistic features than the original attention mechanism. Subsequently, Google's BERT (Devlin et al. 2018) has given way to a whole new family of language models able to produce state-of-the-art contextual embeddings for both individual tokens in a sentence and for the sentence itself. The following paragraphs will explain these architectures in detail.

2.3.1 Basic neural architectures: Artificial neural networks, recurrent networks and convolutional networks

An Artificial Neural Network (ANN) is a mathematical model inspired by the biology of animal brains, though its concept and design have drifted considerably from any kind of biological resemblance. In this section we briefly describe the three basic architectures of ANNs we used to perform sentence classification as part of the task of Fact-extraction: Feedforward Neural Networks (FNN), Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN).

Feedforward Neural Networks (FNN)

FNNs are neural networks where no cyclic structures occur when moving the information forward. It is the simplest model of ANNs.

Let $\mathbf{x} \in \mathbb{R}^n$ be a vector, $\mathbf{W} \in \mathbb{R}^{m \times n}$ a matrix of real-valued parameters, and $\mathbf{b} \in \mathbb{R}^m$ a real-valued vector, the bias. A one-layer FNN f computes the non-linear transformation resulting from the matrix product of x and W, the addition of a bias term, and passing the result through a non-linear activation function g:

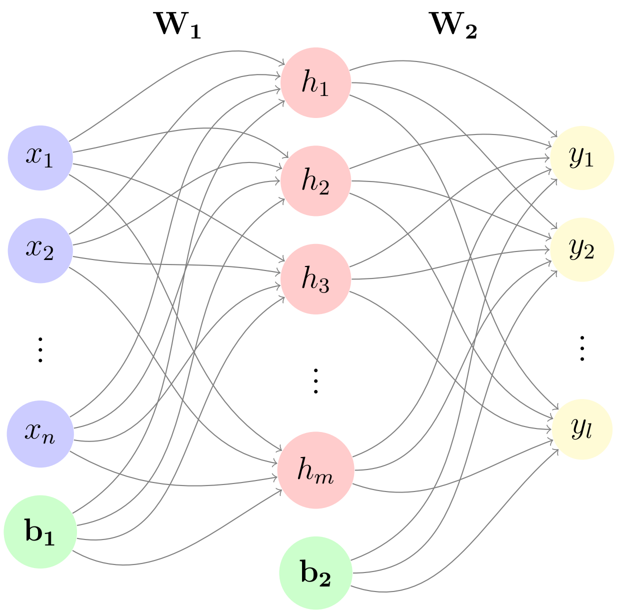


Figure 3: Visual representation of a two-layer feedforward neural network, where $x_1, x_2, \ldots, x_n \in \mathbf{x}$, $h_1, h_2, \ldots, h_m \in \mathbf{h}$, and $y_1, y_2, \ldots, y_l \in f(\mathbf{x})$

$$f(\mathbf{x}) = g(\mathbf{W}\mathbf{x} + \mathbf{b})$$

A multi-layer FNN computes more than one of such transformations:

$$\mathbf{h} = g(\mathbf{W_1}\mathbf{x} + \mathbf{b_1})$$
$$f(\mathbf{x}) = g(\mathbf{W_2}\mathbf{h} + \mathbf{b_2})$$

Where $\mathbf{h}$ is a real-valued vector of arbitrary dimension representing the **hidden layer** of f. Deep networks consist of models with multiple hidden layers. Figure 3 provides a visual representation of a FNN.

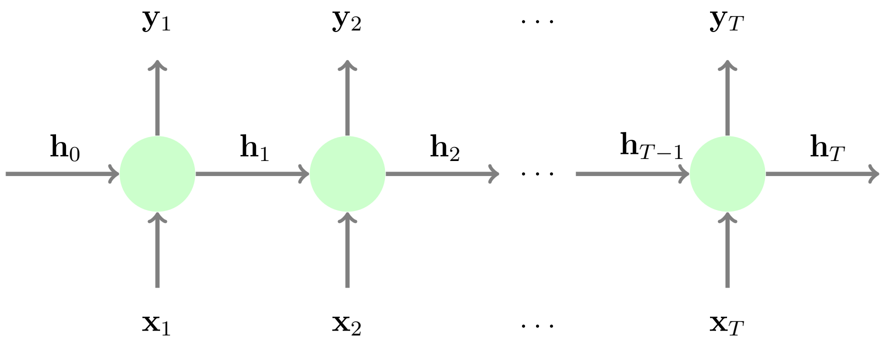


Figure 4: Visual representation of a recurrent neural network. The recurrent unit, in green, is responsible for transmitting the information across time to the next recurrent unit in the sequence.

Recurrent Neural Networks (RNN)

RNNs are neural models designed to take input of a sequential nature. In addition to a similar feed-forward component, RNNs include a **hidden state** vector that makes the output of a token at a given time a function of both its input and the input at the previous step.

The basic form of a RNN is as follows: Let $x = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t)$ be a sequence, and let each $\mathbf{x_i}$ be an n-dimensional vector; let $\mathbf{W_h} \in \mathbb{R}^{m \times n}$, $\mathbf{W_y} \in \mathbb{R}^{l \times p}$ and $\mathbf{U} \in \mathbb{R}^{m \times p}$ be matrices of real-valued parameters; let $\mathbf{b_h} \in \mathbb{R}^m$ and $\mathbf{b_y} \in \mathbb{R}^l$ be bias vectors, and let $\mathbf{h}_t \in \mathbb{R}^p$ be the hidden state at time t. Then, for a RNN f and an activation function g,

$$\begin{aligned}\mathbf{h}_t &= g(\mathbf{W_h x} + \mathbf{U h}_{t-1} + \mathbf{b_h})\\ \mathbf{f}(\mathbf{x})_\mathbf{t} &= g(\mathbf{W_y h}_t + \mathbf{b_y})\end{aligned}$$

where $\mathbf{f}(\mathbf{x})_t$ is the l-dimensional output at time t. Figure 4 shows a visual representation of an RNN.

The basic RNN is seldom used nowadays, having been largely superseded by Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber 1997) and Gated

Recurrent Unit networks (GRU) (Cho et al. 2014). Both models make use of “gates”, that is, linear and non-linear transforms aimed at preserving and discarding information produced in previous states.

LSTM networks were first introduced to address the problem of vanishing gradients, wherein the value of the gradient would become vanishingly small as it was computed during backpropagation. An LSTM unit is comprised of a **cell**, an **input gate**, an **output gate**, and, more recently, a **forget gate**. The name “long short-term memory” comes from the fact that the unit is able to arbitrarily maintain certain values as the information travels through time, unlike the basic RNN where no information can be retained intact once the time step is over. In effect, LSTMs are able to deal with long-distance dependencies such as those that occur in natural language.

Here we present the compact version of the equations and gates involved in a single unit, and Figure 5 presents a visualisation of the interaction of each of these vectors. [1]

$$
\begin{aligned}
\mathbf{f}_t &= g(\mathbf{W_f}\mathbf{x}_t + \mathbf{U_f}\mathbf{h}_{t-1} + \mathbf{b_f}) \\
\mathbf{i}_t &= g(\mathbf{W_i}\mathbf{x}_t + \mathbf{U_i}\mathbf{h}_{t-1} + \mathbf{b_i}) \\
\mathbf{o}_t &= g(\mathbf{W_o}\mathbf{x}_t + \mathbf{U_o}\mathbf{h}_{t-1} + \mathbf{b_o}) \\
\tilde{\mathbf{c}}_t &= g_h(\mathbf{W_c}\mathbf{x}_t + \mathbf{U_c}\mathbf{h}_{t-1} + \mathbf{b_c}) \\
\mathbf{c}_t &= \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tilde{\mathbf{c}}_t \\
\mathbf{h}_t &= \mathbf{o}_t \circ g_h(\mathbf{c}_t)
\end{aligned}
$$

The problem of vanishing gradients and long-distance dependencies has also been addressed with GRU. First introduced in the context of Machine Translation, **GRU networks** achieve performances similar to LSTMs (Chung et al. 2014; Greff et al. 2016) while also

[1] Additionally, we recommend Olah (2015)’s excellent guide to understanding LSTMs.

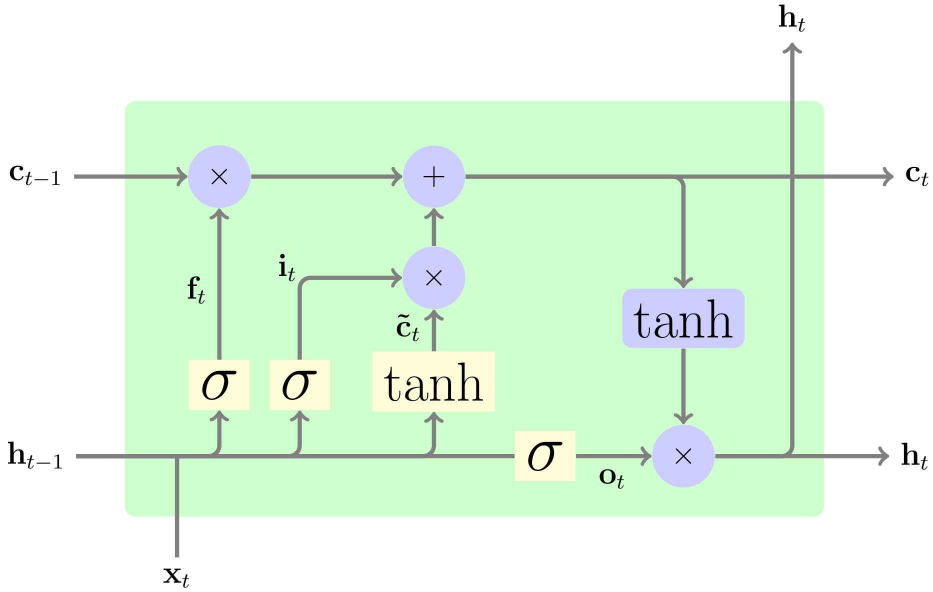


Figure 5: Visual representation of an LSTM unit with a forget gate, the most common variant

being comparatively simpler; the most important modification is the merging of the "forget" and the "input" gates into a single **update** gate. Again, we present the compact equations below and Figure 6 presents a visualisation of the GRU unit.

$$\begin{aligned}
\mathbf{z}_t &= g(\mathbf{W_z} \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t]) \\
\mathbf{r}_t &= g(\mathbf{W_r} \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t]) \\
\tilde{\mathbf{h}}_t &= \tanh(\mathbf{W} \cdot [\mathbf{r}_t * \mathbf{h}_{t-1}, \mathbf{x}_t]) \\
\mathbf{h}_t &= (1 - \mathbf{z}_t) * \mathbf{h}_{t-1} + \mathbf{z}_t * \tilde{\mathbf{h}}_t
\end{aligned}$$

Regardless of the choice of recurrent unit, a recurrent architecture will follow the same design illustrated in Figure 4

Recurrent Neural Networks are useful when dealing with data presented as a time-sensitive sequence. However, they are constrained by the fact that the length of its output

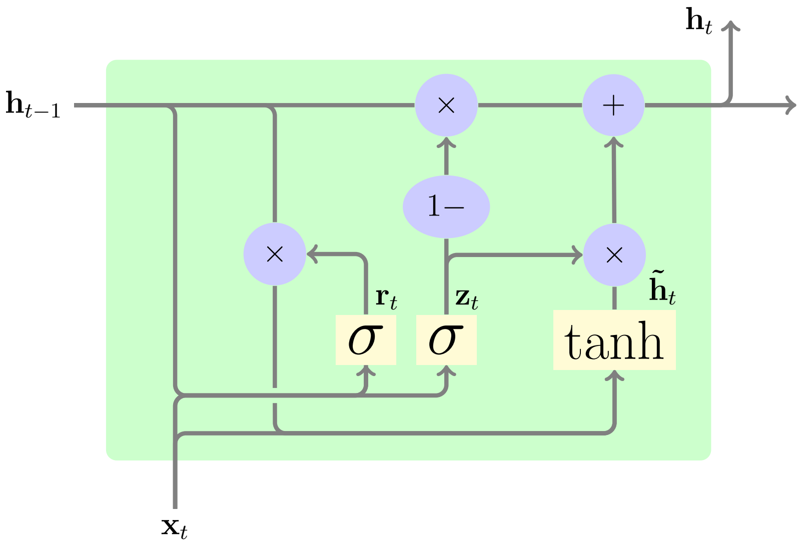


Figure 6: Visual representation of a fully gated GRU unit, the most common variant

must match the length of its input. NLP tasks, such as Machine Translation (ML) require time-sensitive models that are able to handle different lengths between input and output; for example, the Sindarin phrase *Ónen i-Estel Edain* (4 tokens), which is translated as *I gave hope to Men* (5 tokens in English) would be impossible to model using a traditional RNN because of the mismatch between the length of the input and output sequences. In order to work around this constraint, we will use the Encoder-Decoder model.

Encoder-Decoder

To deal with the constraint of having the same dimensionality in both input and output vectors in RNNs, Sutskever, Vinyals, and Le (2014) introduced the Sequence-to-Sequence (seq2seq) model, a recurrent model able to model input and output sequences of different lengths. The seq2seq model uses a novel architecture ensemble called the Encoder-Decoder. The **Encoder** is a model that produces a *context vector* that contains a summary of the entire input sequence; the **Decoder** is a second model, usually a recurrent network, that decodes the context vector and produces a corresponding output sequence, which need not have the same length as the input. Figure 7 shows a visual representation of the Encoder-Decoder.

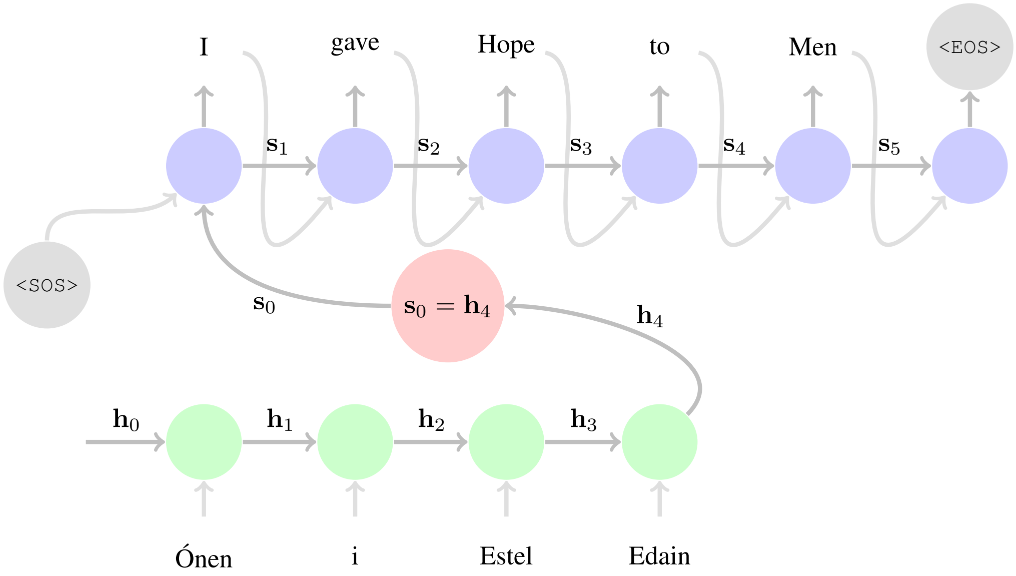


Figure 7: Visual representation of an Encoder-Decoder. The input sequence, of different length than the output, is encoded into a context vector ($\mathbf{h}_4 = \mathbf{s}_0$), which is then decoded as the output sequence.

Several methods of encoding the context vector have been proposed. Figure 7 illustrates the original method proposed by Sutskever, Vinyals, and Le (2014) and Cho et al. (2014) to compute the context, using the final hidden state of the input sequence as the initial hidden state of the output sequence (ie $\mathbf{h}_4 = \mathbf{s}_0$).

Attention

If a long input sequence is processed by an Encoder-Decoder, the intermediate context vector is burdened with having to encode much information into, usually, a fixed-dimension representation. The farther an input token is from the final hidden state that corresponds to the context vector, the less information about this token will be included in the context; moreover, the context vector is usually not enough to capture long-distance dependencies between the tokens in the input. For example, languages with richer inflection than English oftentimes have to keep track of gender and number markers across the nouns of a sentence.

In order to deal with the overburdening of information into a single, fixed-length context vector, Bahdanau, Cho, and Bengio (2014) introduced the Attention mechanism. At any given time step, the decoder uses three elements to produce an output: the previous hidden state, the output from the previous time step, and a weighted representation of the tokens in the input sequence. The weighted representation of the input sequence, called the *annotations* in the original paper, acts as a measure of how important each of the input tokens is for the decoding process at the current time step; in other words, the decoder is told which tokens to pay attention to when producing the current output. Figure 8 shows a visual representation of the mechanism.

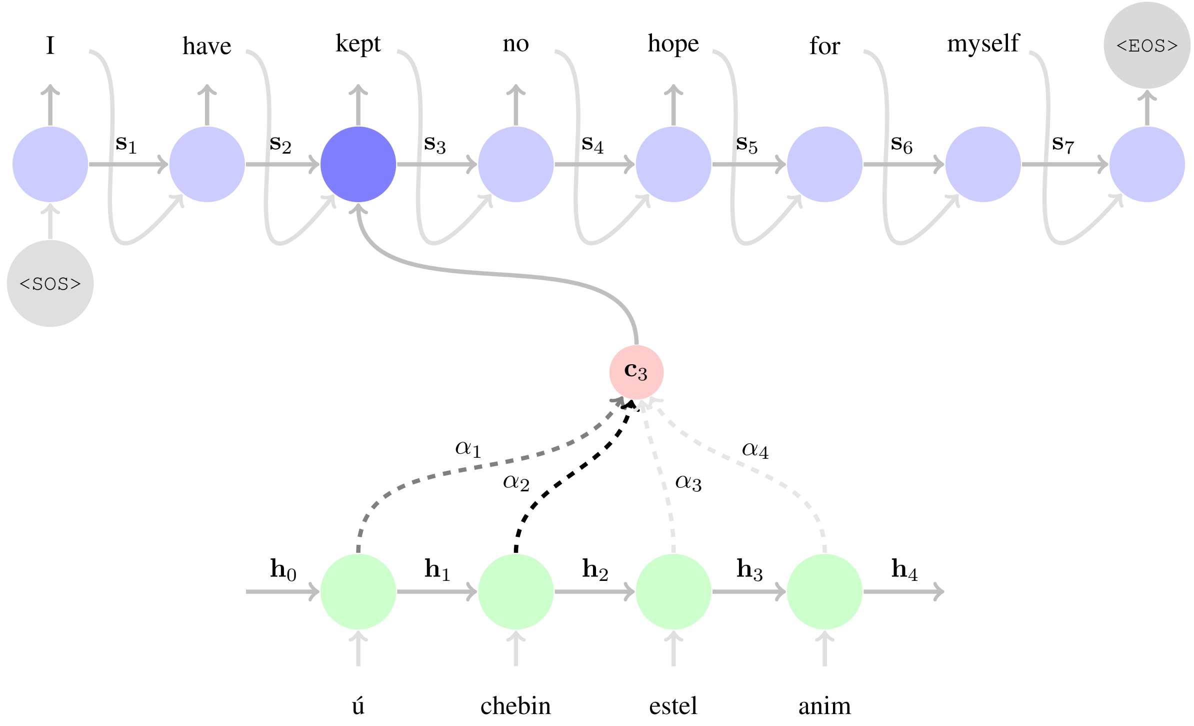


Figure 8: Visual representation of an Encoder-Decoder using an Attention mechanism. At $t = 3$, the word *kept* is produced as the combination of the previous hidden state, the previous output, and the context vector corresponding to the weighted (α_i) annotations of the input sequence.

Convolutional Neural Networks

CNNs are neural models designed to take in data where the context surrounding the input determines the value of the model's output. Originally, CNNs were introduced as means of dealing with visual media. Given a one or two-dimensional input over a number of channels, a number of tensors, each of the same dimensionality, called **kernels**, are swept over the surface of the image, performing the convolution operation over one set of pixels at a time. Each convolution yields a numerical value, and the set of convolutions produced by each kernel yields a **feature map**, an intermediate representation of the image; this can be further subsampled using different methods to produce another representation of the original input. An image can thus be processed by a deep convolutional network, where each layer produces a set of representations by means of convolution operations or subsampling. Figure 9 shows a visual representation of the process.

CNNs can also be used to process text at the token or the character level, by sweeping a kernel over an input sequence and yielding feature map representations of the the text. The resulting representations contain limited, short-range contextual information of each of the tokens or characters in the sequence. Figure 15 shows a visual representation of the process, where the input is a sequence of tokens and the convolution is performed over a fixed-width window of contiguous tokens.

2.3.2 Word embeddings

Tokens cannot be fed directly into a neural model. They require a numerical representation, the most elementary of which is the **one-hot vector** representation: each word in the vocabulary is represented by a sparse binary vector whose length is the size of the vocabulary, and whose only non-zero element corresponds to the positional index of the word in the vocabulary. This method is simple and requires little computing power, but it does not capture any of the linguistic features of a given word, such as part-of-speech tag, case, tense,

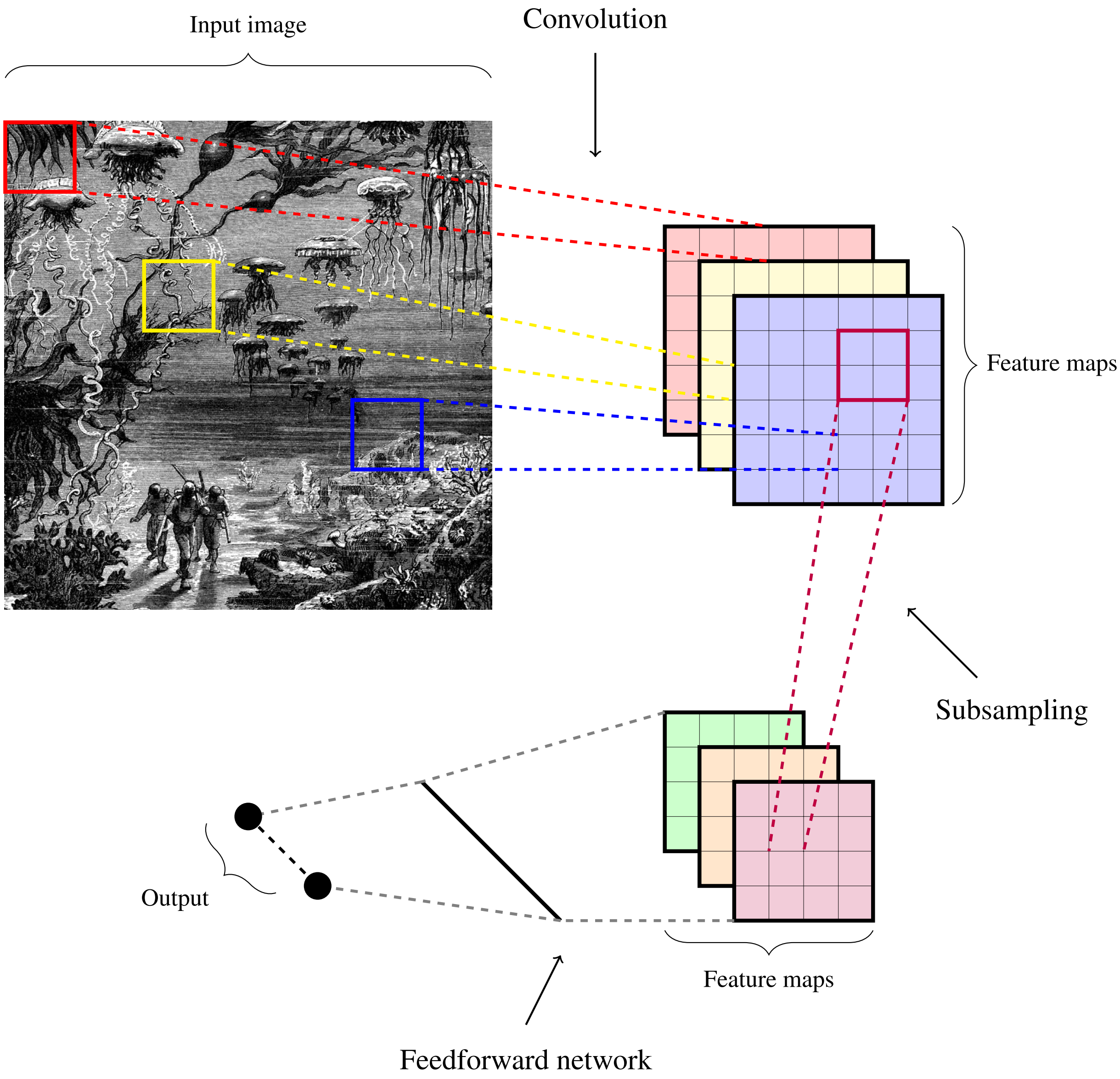


Figure 9: Visual representation of a typical CNN architecture. In this example, the input image is swept by kernels (red, yellow, and blue) to produce the corresponding feature maps. These are subsampled to produce a second set of feature maps, which is passed through a feedforward network in order to produce a binary output.

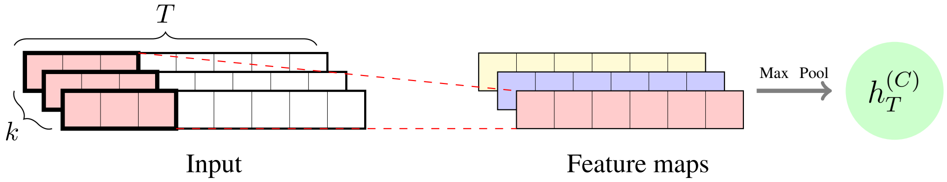


Figure 10: Visual representation of text processing using a CNN. The feature maps produced by the convolutional layers are short-range contextual representations of the input sequences.

inflection, gender, etc. Moreover, the distribution of such vectors in their corresponding vector space offers no insight as to how closely "related" a set of words might be. For example, if the positional indices of words in the vocabulary followed an alphabetical sort, entries such as *hopscotch, Horacio, indulge, intellectual, Jazz*, and *Julio*, which bear no inherent linguistic relation amongst themselves, would all be relatively close to each other.

A better model of lexical vector representation is the **word embeddings** model, introduced by Mikolov et al. (2013). This model produces dense vectors of low dimensionality that, depending on the hyperparamenters of its training, preserve different linguistic features, such as number, gender, and semantics (eg synonyms are usually close to each other). Additionally, arithmetic operations between vectors in the set of embeddings are able to yield relations amongst the words in the vocabulary that a simple one-hot representation would be incapable of capturing. The time-honoured example of such operations is as follows: Let vectors $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3 \in \mathbb{R}^n$ correspond to the words *king*, *man*, and *woman* respectively; then the simple arithmetic expression $\mathbf{w}_1 - \mathbf{w}_2 + \mathbf{w}_3 = \mathbf{x}$ yields a vector in the same vector space; using the metric of *cosine distance* to find the nearest vector in the set, we find such vector, $\mathbf{w}_4$, which corresponds to the word *queen*. In effect,

Example 6.

$$\text{vector}(\text{“king”}) - \text{vector}(\text{“man”}) + \text{vector}(\text{“woman”}) = \text{vector}(\text{“queen”})$$

There are a number of drawbacks in the use of word embeddings. The greatest problem is the relatively large amount of data needed to train them, even though the process itself is *unsupervised*, which means the data on which the vectors are trained need not be annotated with any kind of linguistic information. Another issue is implicit, real-world biases that might exist in the dataset. For example, following the hypothesis of *distributional semantics* (Harris 1954), similar words in a corpus are found surrounded by similar contexts. While Example 6 is famous, a similarly infamous example is shown in Example 7:

Example 7.

$$\text{vector(“doctor”)} - \text{vector(“man”)} + \text{vector(“woman”)} = \text{vector(“nurse”)}$$

Yet another drawback of word embeddings is their difficulty to deal with polysemy: words with identical spellings but different senses can be hard to disambiguate, especially if they share many linguistic features between themselves. For example, the word *plant* is a genderless, singular, and caseless noun which might refer to either the taxonomy of living organisms, a particular member of such taxonomy, or to a human-made structure that generates power. Having a single word embedding for both senses does not distinguish each use.

Despite these hurdles, word embeddings pushed the state-of-the-art in all applications they were used in, and have only recently begun to be replaced by **contextual word embeddings** such as ELMO (Peters et al. 2018) and BERT (Devlin et al. 2018) (see Section 2.3.3). Contextual word embeddings, as their name suggest, replace the unique vector-valued representation of a given word in the vocabulary with a dynamically generated word representation based on the input sentence in which the word is found.

2.3.3 Contextual word embeddings: the Transformer, BERT, and BERT-like models

As an alternative to the original word embeddings model (see Section 2.3.2), in which each word in the vocabulary is assigned a unique, context-independent dense vector, recent approaches to NLP have begun using contextual word and sentence embeddings produced by pre-trained language models such as BERT (Devlin et al. 2018). These have achieved state-of-art in many downstream tasks where the pre-trained model is attached to the relevant architecture and fine-tuned for its specific purposes. Before delving into BERT, however, we must discuss the architecture on which it is based, namely, the Transformer.

The Transformer

The Transformer (Vaswani et al. 2017) is a network architecture proposed to address the recurrent nature of sequence transduction models, ie sequential models that convert an input sequence into an output sequence, such as those used in Machine Translation and Question Answering. The recurrent architecture of these models precludes the use of parallelisation and, as such, oftentimes require higher costs in memory and processing power as the input sequences become longer.

The Transformer proposes a novel way of dispensing of sequential transduction models by means of solely applying the Attention mechanism (see Section 2.3.1) to capture global dependencies between the input and the output. Let $\mathbf{q}, \mathbf{k} \in \mathbb{R}^{d_k}$ and $\mathbf{v} \in \mathbb{R}^{d_v}$ be vectors, called *query, key* and *value*, respectively. The Attention mechanism can be thought of as taking the dot-product of the query and the key, passing it through a softmax layer, and then taking the dot-product of the result and the value vector; in this setup, the query and key are used to calculate the attention weights, which are then applied to the annotations, represented by the value vector. The implementation used in the Transformer generalises this idea into using sets of queries, keys, and values and grouping them into corresponding matrices, while also adding a scaling factor in order to account for the growing magnitude of the dot-product for greater values of d_k.

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{QK^T}}{\sqrt{d_k}}\right)\mathbf{V}$$

This mechanism is called *Scaled Dot-Product Attention* in the original paper, and is the basis for the central component of the Transformer, the *Multi-head Attention* mechanism: each matrix is projected into different, learned linear projections of d_k, d_k and d_v dimensions respectively. These projections are passed through the Attention function, concatenated and finally projected one more time in order to compute the Multi-head Attention.

$$\begin{aligned} \text{head}_i &= \text{Attention}(\mathbf{QW}_i^{(\mathbf{Q})}, \mathbf{KW}_i^{(\mathbf{K})}, \mathbf{VW}_i^{(\mathbf{V})}) \\ \text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= \text{Concat}(\text{head}_i, \ldots, \text{head}_h)\mathbf{W} \end{aligned}$$

The Transformer itself is presented as an Encoder-Decoder model, albeit one that does not use recurrent networks at all. Instead, the Encoder part is a stack of N multi-head attention layers, each coupled with a feedforward network with a residual connection, that yield an output that is passed into the Decoder part, which is another stack of N Multi-head attention layers, each also coupled with a feedforward network. Figure 11 shows a visual representation of the Transformer Encoder and Decoder. [2]

[2] For a more detailed explanation on the inner workings of the Transformer, we recommend the excellent guide *The Annotated Transformer* (Rush, Nguyen, and Klein 2018).

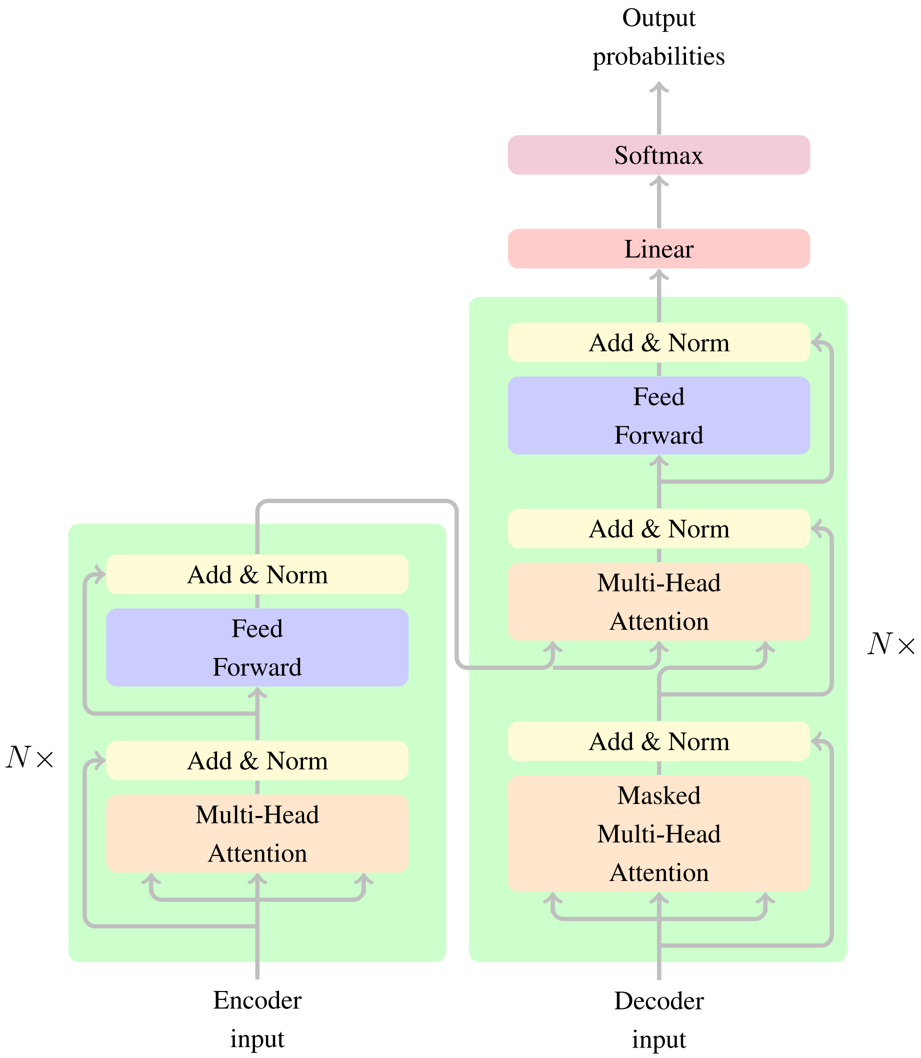


Figure 11: Visual representation of the Transformer. Each feedforward network in the stacks is fed with both the output of the Multi-Head Attention sublayer and a residual connection from the previous layer in the stack. The Encoder part feeds an output into the Decoder, which also contains a Masked Multi-Head Attention layer in order to prevent current positions to attend to subsequent positions.

2.3.4 BERT

BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al. 2018) is a pre-trained language representation model capable of producing richly represented contextual word and sentence embeddings. Its architecture consists of a stack of L bidirectional Transformer encoders trained over an unannotated corpus using two tasks: Masked Language Model (Masked LM) and Next Sentence Prediction (NSP). Masked LM consists of having the model predict randomly masked tokens in its inputs, akin to the cloze procedure (Taylor 1953). NSP consists of a binarised test wherein two sentences, A and B are taken from the corpus and the model determines whether A is followed by B or not.

BERT's greatest advantage is its fine-tuning capabilities. It can be attached to any downstream task and fine-tuned along with all the other task-specific parameters. Fine-tuning consumes much fewer resources than pre-training, and the pre-trained model is widely available, notably as part of the Transformers library (Wolf et al. 2019). BERT has achieved state-of-the-art results in many NLP tasks, including Language Understanding, Question Answering, and Natural Language Inference (Devlin et al. 2018).

2.3.5 RoBERTa and CamemBERT

Several variants of BERT have been produced that modify its architecture and training tasks in order to increase its usability across languages and downstream tasks. One of these variants is RoBERTa (Robustly optimised BERT approach) (Liu et al. 2019), which re-trains the BERT model by performing the following modifications:

1. Using a dynamic masking pattern, in which each masking pattern in the Masked LM is generated every time a new sequence is fed into the model.

2. Removing the NSP training task.

3. Increasing the training batch size.

4. Replacing embedding models as inputs and instead using Byte-Pair Encoding (Sennrich, Haddow, and Birch 2015), where sub-word units replace words, and are extracted by performing statistical analysis on the corpus.

RoBERTa achieved even better performance results than BERT, and itself gave way to a family of BERT-like models, including the one we use in this thesis: **CamemBERT** (Martin et al. 2019), a re-training of RoBERTa on the French part of the OSCAR corpus (Suárez, Sagot, and Romary 2019). Similarly to RoBERTa, camemBERT creates word and sentence representations in the form of contextual word and sentence embeddings but for French.

2.4 Chapter summary

In this chapter we have reviewed previous work on the application of NLP in the legal domain, largely consisting of rule and feature based systems. We have also touched upon the work done in Text Classification and Text Segmentation. Finally, we have introduced the neural methods used in the methodologies proposed in this thesis (see Chapters 4 and 5). The next chapter will present the dataset used to train and evaluate our neural models and segmentation method.

Chapter 3

The dataset

In this chapter, we describe the dataset we used to perform the text segmentation task described in Chapter 1. We describe the way by which the original dataset provided by the *Régie du logement* of the city of Montréal is mined for annotated training instances. After this, we describe the manner in which we build a suitable gold standard. Finally, we describe the process of Part of Speech-based Lexical Substitution Data Augmentation (PLSDA) (Xiang et al. 2020), a technique of data augmentation that we used to increase the amount of training instances in our annotated corpus.

3.1 The original dataset

JusticeBot (Westermann et al. 2019) is an AI tool developed to simplify public access to legal information. It is an interactive system where the user is asked a set of questions about the issue in question and is given information related to previous similar cases. It was developed using a corpus of 667,305 decisions in French, provided by the *Régie du Logement* of the city of Montréal. One of the numerous tasks related to the development and training of the chatbot is the extraction of case-related facts from a given document. Figure 12 shows an example document. Statistics of the original dataset are given in Table 1

Number of documents	667,305
Av number of words	363 ± 434
Number of distinct types	337,491

Table 1: Statistics of the original dataset

(Salaün et al. 2020).

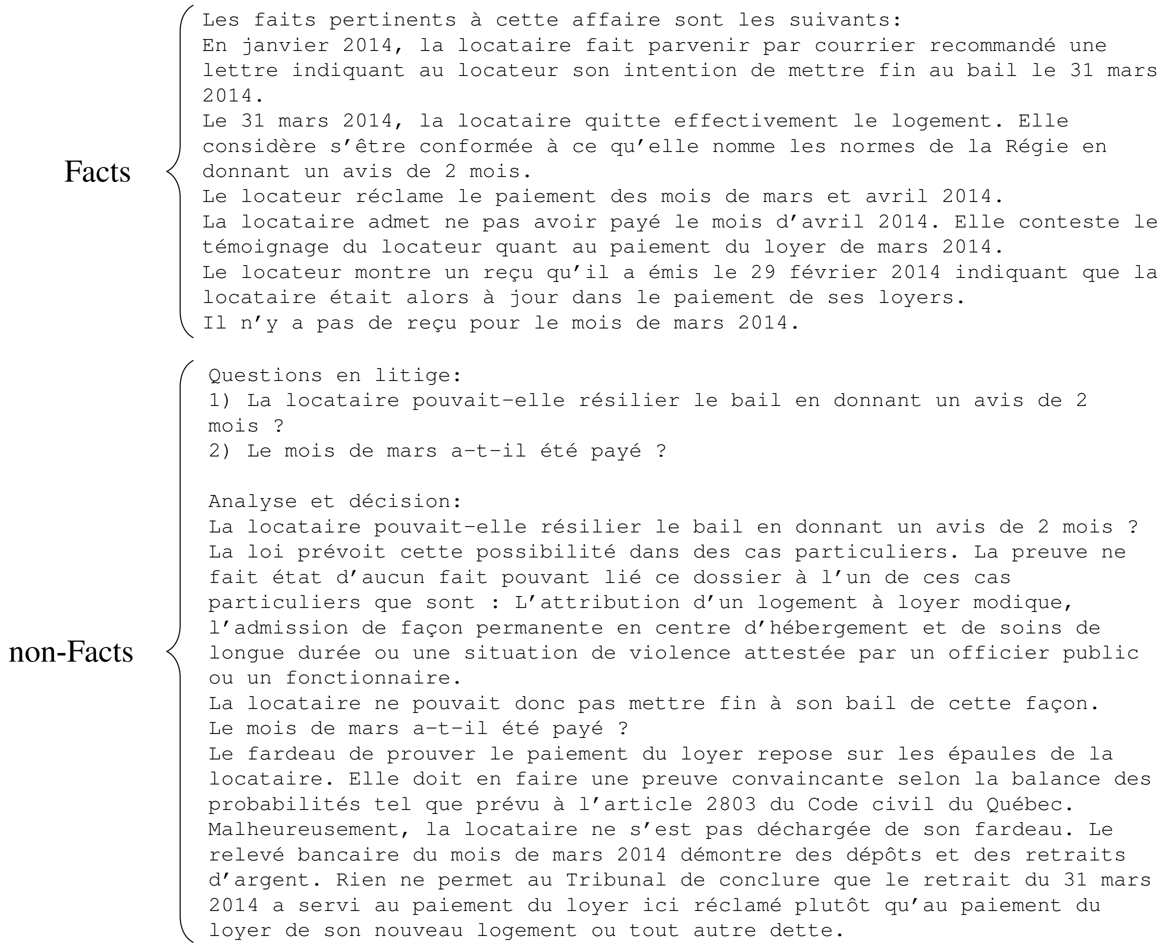
As Figure 12 shows, the original author of the ruling (the judge), naturally delimited Facts and Analysis sections through the use of appropriate headings. However, most of these documents do not clearly separate these two sections. Therefore, our first task consisted in creating an annotated corpus out of this original corpus.

3.2 Extraction of annotated cases

The dataset used for fact extraction consists of a subset of the *Régie du Logement*'s corpus and includes 5,605 unique annotated rulings; these were automatically selected from the original dataset because they include explicit separation between two distinct sections: Facts and Analysis.

In order to extract an annotated corpus from the original, unannotated corpus provided by the *Régie du logement*, we used a heuristic based on regular expressions that matched common headings titles that denoted an explicit separation of case-related facts and judicial analysis. As stated in Section 3.1, the vast majority of documents (99.4%) in the unannotated corpus contain text with no explicit demarcation between the facts that led to the judicial decision and the analysis and application of these; and those that did use such demarcation follow an irregular pattern, with some using headings and some simply using paragraph breaks.

We developed regular expressions to extract the annotated documents based on the


Facts
non-Facts

Les faits pertinents à cette affaire sont les suivants:

En janvier 2014, la locataire fait parvenir par courrier recommandé une lettre indiquant au locateur son intention de mettre fin au bail le 31 mars 2014.

Le 31 mars 2014, la locataire quitte effectivement le logement. Elle considère s’être conformée à ce qu’elle nomme les normes de la Régie en donnant un avis de 2 mois.

Le locateur réclame le paiement des mois de mars et avril 2014.

La locataire admet ne pas avoir payé le mois d’avril 2014. Elle conteste le témoignage du locateur quant au paiement du loyer de mars 2014.

Le locateur montre un reçu qu’il a émis le 29 février 2014 indiquant que la locataire était alors à jour dans le paiement de ses loyers.

Il n’y a pas de reçu pour le mois de mars 2014.

Questions en litige:

1) La locataire pouvait-elle résilier le bail en donnant un avis de 2 mois ?

2) Le mois de mars a-t-il été payé ?

Analyse et décision:

La locataire pouvait-elle résilier le bail en donnant un avis de 2 mois ?

La loi prévoit cette possibilité dans des cas particuliers. La preuve ne fait état d’aucun fait pouvant lié ce dossier à l’un de ces cas particuliers que sont : L’attribution d’un logement à loyer modique, l’admission de façon permanente en centre d’hébergement et de soins de longue durée ou une situation de violence attestée par un officier public ou un fonctionnaire.

La locataire ne pouvait donc pas mettre fin à son bail de cette façon.

Le mois de mars a-t-il été payé ?

Le fardeau de prouver le paiement du loyer repose sur les épaules de la locataire. Elle doit en faire une preuve convaincante selon la balance des probabilités tel que prévu à l’article 2803 du Code civil du Québec.

Malheureusement, la locataire ne s’est pas déchargée de son fardeau. Le relevé bancaire du mois de mars 2014 démontre des dépôts et des retraits d’argent. Rien ne permet au Tribunal de conclure que le retrait du 31 mars 2014 a servi au paiement du loyer ici réclamé plutôt qu’au paiement du loyer de son nouveau logement ou tout autre dette.

Figure 12: Example of a document from the original dataset. *Facts* and *non-Facts* sections are separated by headings.

Number of documents	5,605
Total number of sentences	454,210
Total number of sentences in *Facts* segments	239,077
Av number of sentences in *Facts* segments	36.25
Av number of words in *Facts* segments	832.08
Total number of sentences in *non-Facts* segments	215,133
Av number of sentences in *non-Facts* segments	32.62
Av number of words in *non-Facts* segments	803.74

Table 2: Statistics of the annotated dataset used in our work.

most common headings that unambiguously denoted explicit demarcation between *Facts* and *non Facts*. We experimented with several different combinations of common phrases in order to take advantage of decisions using paragraph breaks as delimiters, but we consistently obtained "noisy" datasets, that is, datasets containing several hundreds of instances where unexpected formatting styles (eg arbitrary capitalisation, unexpected punctuation, unexpected itemisation, etc) would yield documents whose segmentation was not clear at all. Since precision was more important than recall, we preferred to have a less noisy annotated corpus (albeit smaller) than a larger one that may be more noisy. Hence, we chose to keep only two phrases that unequivocally yielded easily segmented documents; these were *Les faites pertinents* ("The pertinent facts") and *Analyse* ("Analysis"). Using these as the basis of our regular expressions, we obtained 6,595 annotated documents (0.66% of the original corpus); after removing duplicates, we were left with 5,605 documents. The small fraction of annotated documents is indicative of the problem described in Section 1.4.

Statistics of the resulting annotated dataset are shown in Table 2. As the table shows, the dataset we obtained includes 454,210 sentences annotated as either Facts or non-Facts, split approximately 55-45% respectively. As Table 2 shows, Fact and non-Fact segments are similar both in terms of average number of words per sentence and average number of sentences per segment.

3.3 The gold standard

In order to evaluate the proposed segmentation methods in this thesis, they must be evaluated against a test set that meets two requirements:

1. The segment containing the *Facts* is labelled and clearly demarcated from the rest of the sentences.

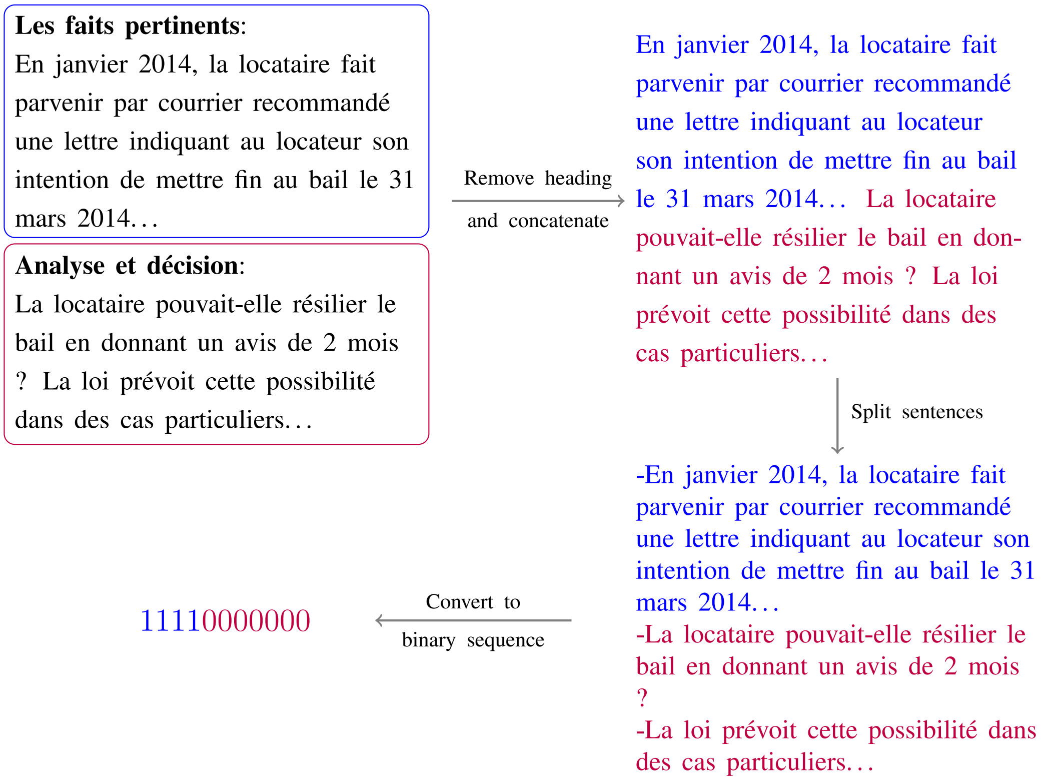
2. The dataset only contains a sequence of sentences without headings to distinguish the *Facts* from the *non-Facts*. Otherwise, the segmentation would be trivial.

Since the annotated corpus we extracted complied with the first condition but not with the second, we further processed it to create a test set that would meet both criteria; this test set will also be called the *gold standard* for the rest of this thesis.

To create the gold standard, each annotated document was stripped from its headings, and the text contained in each section was concatenated to form a continuous sequence of sentences. As Figure 13 shows, we remove line breaks, dashes, and semicolons, replacing them with full stops. We then split the text into sentences using the spaCy NLP library (Honnibal et al. 2020). Finally, we associate each resulting paragraph to a binary sequence in which each digit corresponds to a sentence which is either found in the *Facts* (1) or the *non-Facts* (0) creating a binary tensor of length equal to the number of sentences in the document. The resulting binary sequences serve as targets for the different approaches we followed to perform the text segmentation.

3.3.1 Limitations of the gold standard

The manner in which the gold standard is created is non-ideal for one important reason: The set of documents retrieved from the original corpus deemed to be annotated, as detailed in Section 3.2, is limited by our ability to craft relevant regular expressions that select



Les faits pertinents:
En janvier 2014, la locataire fait parvenir par courrier recommandé une lettre indiquant au locateur son intention de mettre fin au bail le 31 mars 2014…

Analyse et décision:
La locataire pouvait-elle résilier le bail en donnant un avis de 2 mois ? La loi prévoit cette possibilité dans des cas particuliers…

En janvier 2014, la locataire fait parvenir par courrier recommandé une lettre indiquant au locateur son intention de mettre fin au bail le 31 mars 2014… La locataire pouvait-elle résilier le bail en donnant un avis de 2 mois ? La loi prévoit cette possibilité dans des cas particuliers…

-En janvier 2014, la locataire fait parvenir par courrier recommandé une lettre indiquant au locateur son intention de mettre fin au bail le 31 mars 2014…
-La locataire pouvait-elle résilier le bail en donnant un avis de 2 mois ?
-La loi prévoit cette possibilité dans des cas particuliers…

Figure 13: Creation of the gold standard. To create an instance, we remove line breaks, dashes, and semicolons and replace them with headings; we then remove the headings and concatenate the segments; we then split the sentences, and, finally represent the text as a binary sequence, where the Facts sentences are represented as a subsequence of 1's and the non-Facts sentences as a subsequence of 0's.

relevant samples. There might exist documents which we could readily segment into facts and non-facts that were not retrieved by our method, which was reliant on explicit headings appearing in the text. Nevertheless, we considered that such method was the best option to ensure the retrieval of documents which could be automatically segmented with little ambiguity.

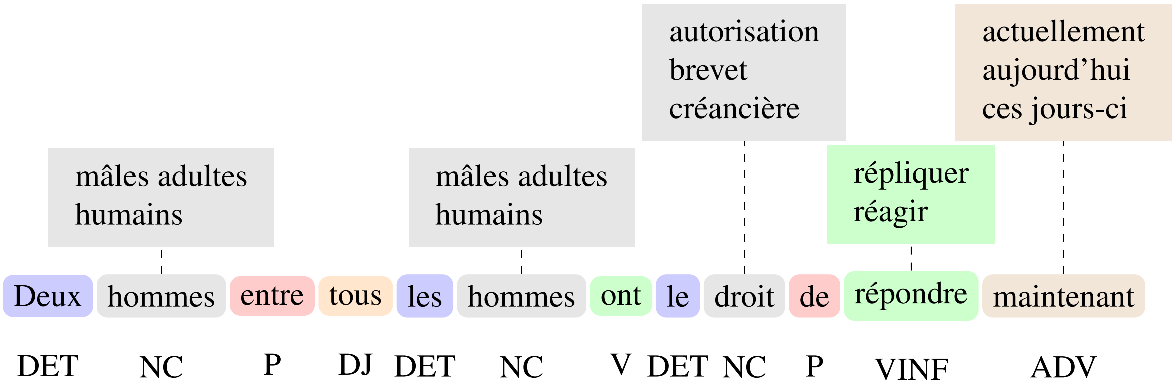


Figure 14: Example of the listing of substitute candidates for words in a sentence. Each word in the sentence is tagged with its POS tag and associated with a list of substitute candidates mined from WordNet. A candidate is probabilistically chosen to create a new sentence that is syntactically identical and semantically similar to the original sentence.

3.4 Data augmentation

In order to account for the relatively low number of training instances in our annotated corpus, we turned to a data augmentation technique called POS-based Lexical Substitution Data Augmentation (PLSDA). Introduced by Xiang et al. (2020), the principle behind PLSDA is simple: Given an input sentence, a list of substitute candidates is created for each word in the sentence, based on the corresponding Part-of-Speech (POS) tag of the word, and using tools such as WordNet (Miller 1995) to obtain a list of syntactically equivalent synonyms. Using a probabilistic approach to select the substitutions, a new training instance is generated each time a substitution occurs in the original training instance.

For example, consider the phrase *Deux hommes entre tous les hommes ont le droit de répondre* ("Two men, amongst all men, have the right to answer."). Each of the words in the sentence is tagged with its corresponding POS tag, as indicated in Figure 14, and using this information, a set of synonyms is extracted from WordNet for each of the annotated tokens. Because of syntactic consistency, if we were to replace any token with another, identically tagged token (eg replace *hommes*, "men", with *mâles adultes*, "adult males"),

Final datasets	1	2	3
Number of documents	5,605	11,210	16,815
Total number of sentences	454,210	838,347	1,257,519
Total number of sentences in Facts segments	239,077	442,481	717,231
Av number of sentences in Facts segments	36.25	36.25	36.25
Av number of words in Facts segments	832.08	832.08	832.08
Total number of sentences in non-Facts segments	215,133	395,866	593,799
Av number of sentences in non-Facts segments	32.62	32.62	32.62
Av number of words in non-Facts segments	803.74	803.74	803.74

Table 3: Statistics of the dataset augmented using PLSDA. “Augment” refers to the factor by which the dataset was increased.

the resulting sentence would be syntactically identical to the original; additionally, because of semantic consistency, if we were to replace a token with a synonym with an identical POS tag, the resulting sentence will have similar semantics to the original sentence, thus rendering it a potential training instance for our task.

Because of the relative scarcity of resources in the French section of multilingual WordNet, we modified the procedure used by Xiang et al. (2020) and selected only nouns and adjectives as candidates for substitutions. Nevertheless, we were able to approximately double and triple the number of documents and the number of sentences in our dataset. The selection of the substitution candidates was done as follows: for each sentence in dataset, the token with the most synonyms was the one selected for substitution, and the synonym was randomly selected from the associated synset following a uniform distribution. The substitution was carried out only one token at a time per sentence.

3.5 Chapter summary

In this chapter we described the original corpus from the *Régie du Logement* that we used in this thesis and how we mined from it the annotated training instances in order to build an annotated corpus with which we could train and evaluate our models. We also described the process by which we build the appropriate gold standard with which we evaluate the segmentation task. Finally, we introduced and described the PLSDA technique (Xiang et al. 2020), which we applied to our annotated corpus in order to increase the number of training instances.

In the next chapter, we will describe the first approach we developed to segment legal texts into *Facts* and *non-Facts*.

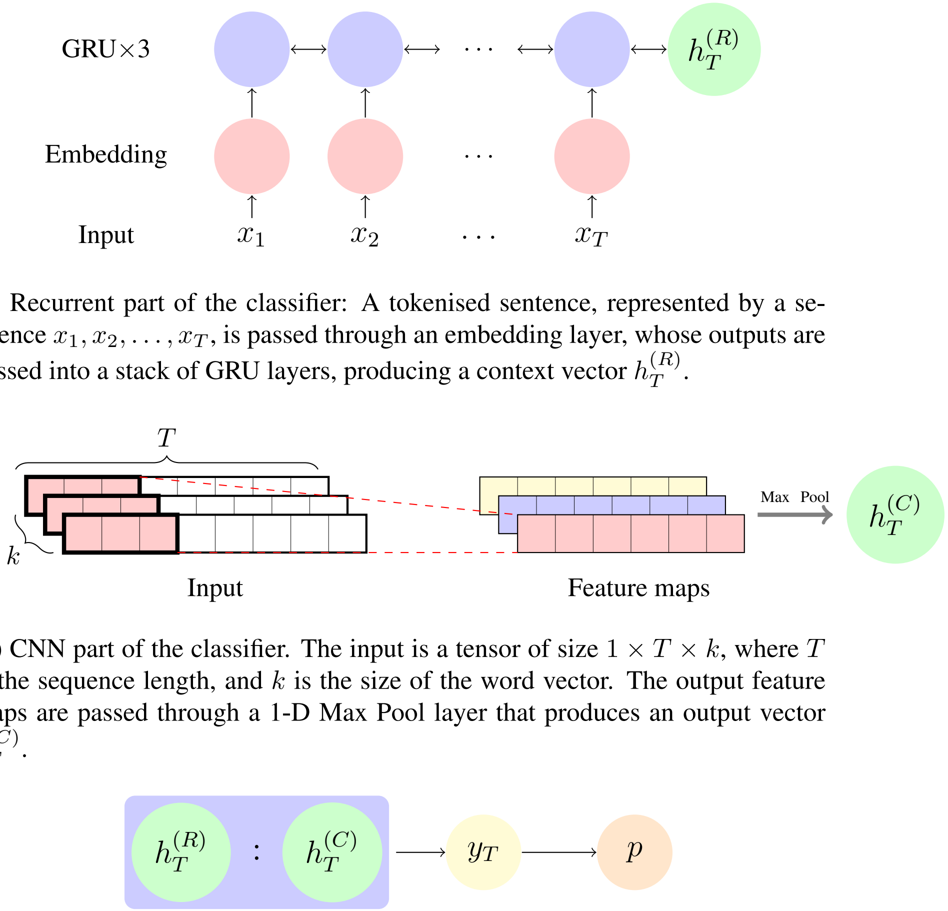
Chapter 4

Text segmentation using sentence classification

This chapter will describe our first approach to perform text segmentation in order to extract Facts from the documents of our corpus. The approach takes as input a sequence of sentences and produces a binary sequence, where each sentence is classified either as *Facts* sentence or a *non-Facts* sentence. This binary classification approach would ideally result in a representation where all the sentences belonging to the facts will appear as a contiguous sequence of 1's and all other sentences will appear as a contiguous sequence of 0's. The evaluation of this approach (see Sections 4.2 and 4.3.1) show that this method, while yielding a performance that surpasses that of the baseline, can be further improved by taking into account the context in which each sentence is found.

4.1 Sentence classification

Our first approach at Fact segmentation is based on the binary classification of each sentence, where only the content of the of the sentence to classify is taken into account.



(a) Recurrent part of the classifier: A tokenised sentence, represented by a sequence $x_1, x_2, \ldots, x_T$, is passed through an embedding layer, whose outputs are passed into a stack of GRU layers, producing a context vector $h_T^{(R)}$.

(b) CNN part of the classifier. The input is a tensor of size $1 \times T \times k$, where T is the sequence length, and k is the size of the word vector. The output feature maps are passed through a 1-D Max Pool layer that produces an output vector $h_T^{(C)}$.

(c) Concatenation of the context vector $h_T^{(R)}$ and the output vector $h_T^{(C)}$. The resulting tensor is passed through an affine layer with a softmax activation function to produce the probability of the sentence being Facts.

Figure 15: Architecture of the sentence classifier.

4.1.1 Model architecture

To perform the sentence classification, each sentence in the text is converted to a tensor using a vocabulary and a list of indices built from the pre-trained, non-domain-specific, French word embeddings model of Baroni et al. (2009). The tensor is then fed into a neural model that outputs a binary classification, indicating whether the sentence belongs to the facts or the non-facts. The architecture of the sentence classification is shown in Figure 15.

The model is a combination of recurrent and convolutional architectures. Overall, a sequence of sentences are fed through the model, and a binary sequence of L entries is produced, where the value 1 represents sentences that belong to the facts and 0 represents those belonging to the non-facts. In the recurrent architecture (see Figure 15a), a tokenised sentence of length T, represented by a sequence of words $x_1, x_2, \ldots, x_T$, is passed through an embedding layer, whose outputs are passed into a bidirectional stack of Gated Recurrent Unit (GRU) layers (Sutskever, Vinyals, and Le 2014), producing a context vector $h_T^{(R)}$. In the convolutional part (see Figure 15b), the input is a tensor of size $n \times T \times k$, where n is the batch size and k is the size of the word vector (ie the embedding size). The output feature maps are passed through a 1-D Max Pool layer that produces an output vector $h_T^{(C)}$. Finally, $h_T^{(R)}$ and $h_T^{(C)}$ are concatenated (see Figure 15c), and the resulting vector is passed through an affine layer with a softmax activation function to produce the probability of the sentence being a *Fact*.

We used both the original dataset and the augmented datasets detailed in Section 3.4. We used 75% of the dataset for training, 10% validation, and the remaining 15% for testing. The model's hyperparameters are shown in Table 4.

General hparams		**GRU hparams**	
Batch size (n)	512	Num layers	3
Embedding size (k)	200	Dropout	0.5
Gradient clipping	0.2	Hidden size	512
Weight decay	1e-4	Bidirectional	Yes
Learning rate	1e-{3,4,5}		
Optimiser	ADAM	**CNN hparams**	
		Dropout	0.2
		Output channels	100
		Kernel size	3

Table 4: Model hyperparameters of the sentence classifier

4.2 Evaluation of the sentence classifier

We evaluated the sentence classifier using the test datasets of both the original corpus and the augmented corpora (15% of each). As the method of intrinsic evaluation, we used accuracy, precision, recall and the F_1 score for the class *Fact*, using the batch size as sample size. Results are shown in Table 5.

	Accuracy	Precision	Recall	F_1
Original dataset	0.7887	0.7770	0.7569	0.7668
Augmented dataset by 2	0.7906	0.7797	0.7579	0.7686
Augmented dataset by 3	0.7886	0.7773	0.7562	0.7668

Table 5: Evaluation metrics of the sentence classifier using both the original and the augmented datasets

4.3 Text segmentation

Once each sentence is classified as fact (1) or non-fact (0), the next step is to optimally divide the sequence into two substrings, each representing the *Facts* and *non-Facts* segments of the ruling. Figure 16 illustrates this. To perform the segmentation, we establish the following propositions:

- Let L represent the number of sentences document and let L_f represent the number of sentences in its Facts segment.
- Let n_f represent the number of Facts sentences found in L_f (such that $n_p \leq L_f$).
- Define $p_f = \frac{n_f}{L_f}$ as the *purity* of L_f.

We wish to maximise both L_f and p_f; maximising L_f is equivalent to maximising the recall of facts in the segmentation, and maximising p_f is done to ensure the segmentation

$$\underbrace{\overbrace{1111011111111}^{L_f}00000100001}_{L}$$

Figure 16: Visual representation of the segmentation of the binary string where 1 refers to a sentence classified as *fact* and 0 as *non-fact*. In this example, $L_f = 13$, $L = 24$ and $n_f = 12$

.

corresponds as closely as possible to the gold standard, in keeping with our central assumption (see Figure 2). Maximising both variables at the same time can be described as the following optimisation problem:

$$\max J(L_f) = \max\left(\alpha L_f + \beta p_f\right) \tag{1}$$

where J is a loss function of L_f, and $\alpha, \beta \in \mathbb{R}$ are arbitrary weights representing the importance of each term. We can rewrite and differentiate Equation 1 to find an expression that optimises L_f:

$$\begin{aligned}
\min -J(L_f) &= \min -\left(\alpha L_f + \beta \frac{n_f}{L_f}\right) \\
-J'(L_f) &= -\alpha + \beta \frac{n_f}{L_f^2} \\
0 &= -\alpha L_f^2 + \beta n_f \\
L^2 &= \frac{\beta}{\alpha} n_f \\
L_f &= \frac{\beta}{\alpha} p_f \qquad (2)
\end{aligned}$$

Equation 2 indicates that there is a linear relation between the purity (p_f) of a substring and its length (L_f). Rather than conducting a large number of experiments to empirically determine optimal values for α and β, we turned to a simpler strategy: given the linear relation between length and purity, for all possible substrings of length L_{f_i} in the original string, we select the one that maximises p_f. Since any substring, including one with a single member, comprised exclusively of 1's will have a trivial purity $p_{f_i} = 1$, we select L_{f_i} that maximises p_f such that $p_f \neq 1$.

By considering the problem of finding the optimal segmentation point as extracting the substring with the highest purity from a binary representation of the document, our model can compute a splitting index, l, which can be empirically weighted by a factor, $\gamma \propto \frac{\beta}{\alpha}$ in order to favour either shorter or longer substrings. Shorter substrings will be purer and will contain few instances of non-Facts, while longer substrings will favour the recall of sentences containing facts. Hence, we can compute the weighted splitting index L_γ as $L_\gamma = \gamma l$

4.3.1 Evaluation of the segmentation task

As a method of extrinsic evaluation, we test the text segmentation over the test fraction of annotated cases in the original corpus (660 documents) using the gold standard described in Section 3.3 and using three classifiers trained respectively on the test set, and the test set augmented by factors of 2 and 3 using the methodology described in Section 3.4. We made sure that no augmented instances were included in the test set.

Given a value of γ, we compute a corresponding splitting index l for each document, and split the text according to the weighted splitting index L_γ. We then count the number of sentences by which the resulting text is off compared to the gold standard of number of sentences in the Facts section. Results are shown in Table 6 and Figure 17 for the original dataset, and in Table 7 and Figure 18 for the augmented datasets.

Using the sentence classifier trained on the original dataset, Table 6 and Figure 17 show that, with the original test, for $\gamma = 1$, we obtain 39 documents ($15 + 11 + 13 = 39$ or 6% of the dataset) that fall within a single sentence of difference with respect to the gold standard, and 11 documents (1.7%) being segmented exactly where the gold standard

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
	0.6	602	25	16	8	3	3	2	0	1	0	0
	0.8	570	25	30	7	13	6	1	3	3	0	2
γ	1	**529**	19	26	21	15	11	13	8	3	5	10
	1.2	525	21	24	22	16	9	11	8	6	8	10
	1.4	520	17	26	21	16	11	13	6	9	7	14

Table 6: Sentence offset results for the different values of γ using the original dataset for training and testing.

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
	0.6	605	19	15	11	3	1	3	1	0	2	0
	0.8	577	24	22	12	8	8	1	2	3	1	2
γ	1	537	19	24	24	15	11	6	7	0	2	15
	1.2	531	22	24	23	15	10	6	10	2	2	15
	1.4	524	20	21	26	18	10	5	9	6	5	16

Table 7: Sentence offset results for the different values of γ using the doubled dataset for training and testing.

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
	0.6	604	21	15	9	6	2	2	1	0	0	0
	0.8	575	26	26	9	10	6	3	1	4	0	0
γ	1	530	22	25	27	18	7	7	5	3	3	13
	1.2	526	24	23	27	18	11	3	6	5	3	14
	1.4	518	19	30	21	24	8	10	5	5	4	16

Table 8: Sentence offset results for the different values of γ using tripled

indicates. Nevertheless, 529 documents (80%) are split using an underestimated index and fall short of the target, being more than 4 sentences away from the gold standard. Increasing the value of γ favours the recall of sentences annotated as Facts, but the percentage of documents whose segmentation falls short by more than 4 sentences does not decrease as quickly as the percentage of overestimated segmentations; for $\gamma = 1.4$, the number of underestimated segmentations by a margin greater than 4 is still 520 (78%).

When using the classifiers trained on the augmented datasets, little to no improvement is observed. Table 7 and Figure 18 show that, with the dataset augmented by a factor of 2, for $\gamma = 1$, we get 32 documents ($15 + 11 + 6$ or 4.8% of the dataset) that fall within a single sentence of difference with respect to the gold standard. However, 537 documents (81%) are still more than 4 sentences away from the expected splitting point. Once again, increasing the value of γ has the same effect as increasing the recall of sentences annotated as Facts, and for $\gamma = 1.4$ we see now that 524 documents (79%) lie more than 4 sentences away from the predicted splitting index.

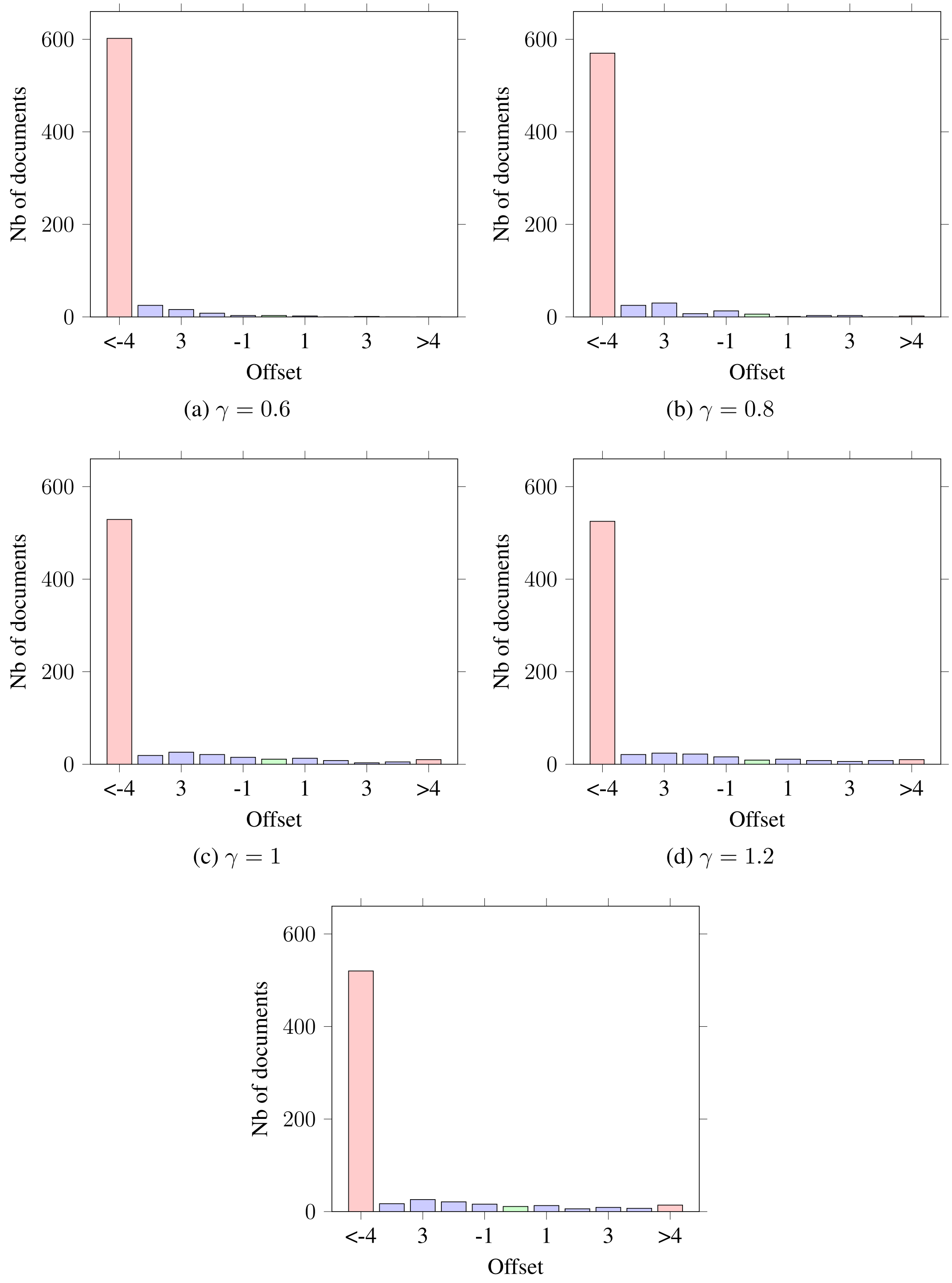
For the different values of γ, the distributions of offsets (Figures 17 and 18) present a large number of underestimated splitting indices in all cases. This fact seemingly stands in contrast to our base assumption (Section 1.2), that facts should always give way to analyses: the gold standard expects us to find many more facts after the predicted splitting index, weighted or otherwise, which suggests that some cases either contain an imbalance of facts and analyses or contain facts and analysis interspersed with each other on a larger scale than expected.

Moreover, the lack of improvement in performance of the sentence classifier using the augmented dataset compared to the original dataset (see Table 5) influences the value of the performance of our splitting heuristic, but it does not significantly alter the trend observed in the charts (see Figures 17 and 18). We speculate that larger amounts of data would not significantly alter the trend, and that only an increase in performance would avoid the considerable percentage of underestimated documents.

4.4 Chapter summary

In this chapter we have described our first approach to Fact segmentation based on the classification of each sentence as either *Facts* or *non-Facts* and segmenting the document according to the greatest number of facts in the longest segment of text. We have shown that the independent classification of sentences as either Fact of non-Fact, and the subsequent segmentation of the resulting binary string, yields a large percentage of documents whose segmentation point is underestimated.

In the next chapter we will describe a different approach, namely, one where the classification of sentences as *Facts* or *non-Facts* is not independent with one another but actually conditioned by the context in which they are found.



(a) $\gamma = 0.6$

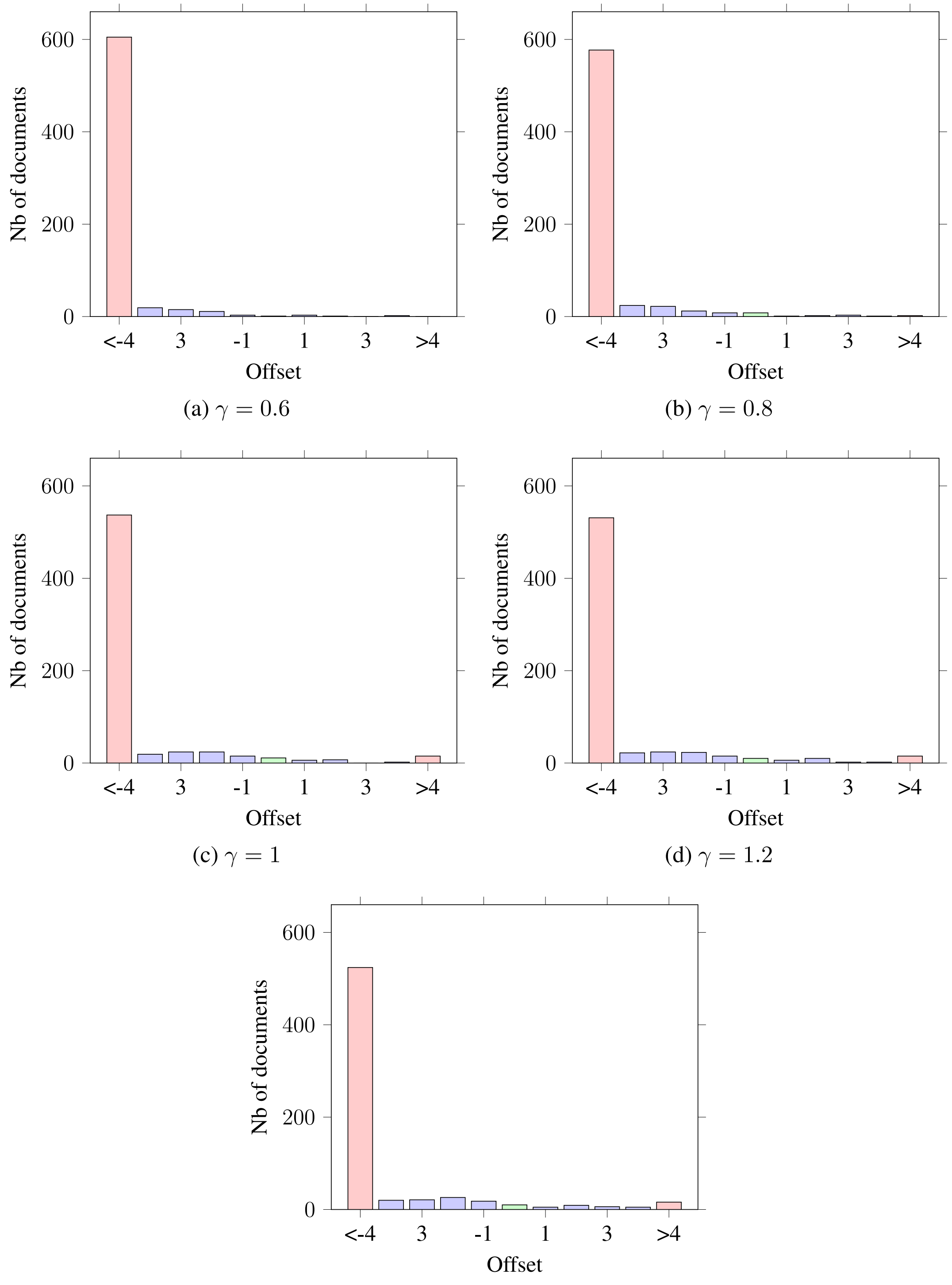
(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure 17: Sentence offset distribution for the different values of γ using the original dataset



(a) $\gamma = 0.6$

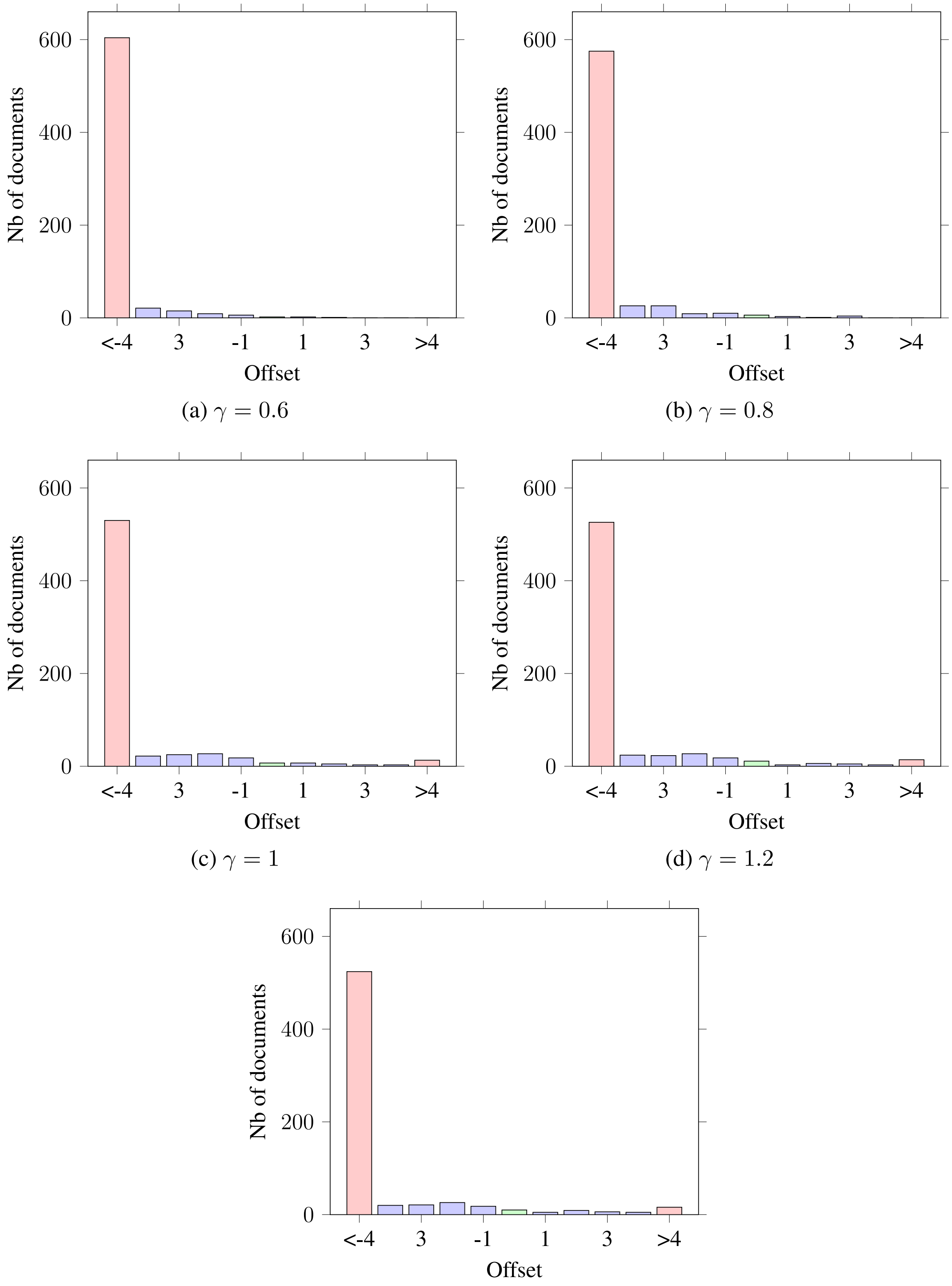
(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure 18: Sentence offset distribution for the different values of γ using the doubled augmented dataset



(a) $\gamma = 0.6$

(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure 19: Sentence offset distribution for the different values of γ using the tripled augmented dataset

Chapter 5

Text segmentation using recurrent architectures

The previous chapter described how to segment the text of a document into *Facts* and *non-Facts* by independently classifying each sentence. This idea relies on the assumption that a sentence will be classified into either category regardless of the context in which it is found. This is a questionable notion at best because of the expected discourse structure of legal documents (see Section 1.2); according to the IRAC paradigm (see Section 1.2, sentences will tend to keep congruent semantics, and the probability of finding a sentence stating a fact amongst sentences discussing an application of the facts is small.

In order to take into account both the content and the context of a given sentence, we turn to a different architecture as the basis of our second approach: a bidirectional recurrent model in which each element in the time series is a function of all others. We represent each document in our dataset as a sequence of sentences, and each sentence as a dense vector, using camemBERT (Martin et al. 2019) to create contextual sentence embeddings. We then experiment with two neural architectures: a recurrent model using an LSTM/GRU recurrent network (Hochreiter and Schmidhuber 1997; Cho et al. 2014) and a sequence-to-sequence (seq2seq) attention model (Bahdanau, Cho, and Bengio 2014).

Section 5.1 will describe the use of Contextual Sentence Embeddings in our models and the memory constraints we had to overcome in order to add them to our architecture. Section 5.2 will describe the specific architecture of the recurrent models we used to carry out the classification task: the LSTM/GRU networks and the Attention model; Section 5.3 will deal with the intrinsic evaluation of these models. Finally, Section 5.4 will present and discuss the results of the evaluation of the segmentation task using recurrent architectures.

5.1 Sentence embeddings

During the course of our experiments, it became apparent that the memory constraints of using a large BERT-like model such as camemBERT would make it impossible to include it as an additional module to fine-tune; because of the average number of sentences per document in our corpus (see Table 2), creating the contextual sentence embeddings as part of the architecture's pipeline exceeded our resources for the vast majority of instances in our training set. Therefore, we had to resort to using camemBERT to pre-train the contextual sentence embeddings without the possibility of fine-tuning the model to our specific task.

5.2 Architectures of the LSTM/GRU networks and the Attention model

We experimented with two recurrent architectures. The first architecture, shown in Figure 20 is based on LSTM and GRU recurrent networks. At each time step, a sentence embedding is fed into the cell and a binary output is produced as a function of the hidden states from both the previous and the following time steps, as well as from the input. The recurrent network is bidirectional, meaning that each output contains information from both previous and future representations of the document. This was meant to add contextual information to the final binary output, which results in a sentence's classification being a function of both its own representation and the context in which it is found.

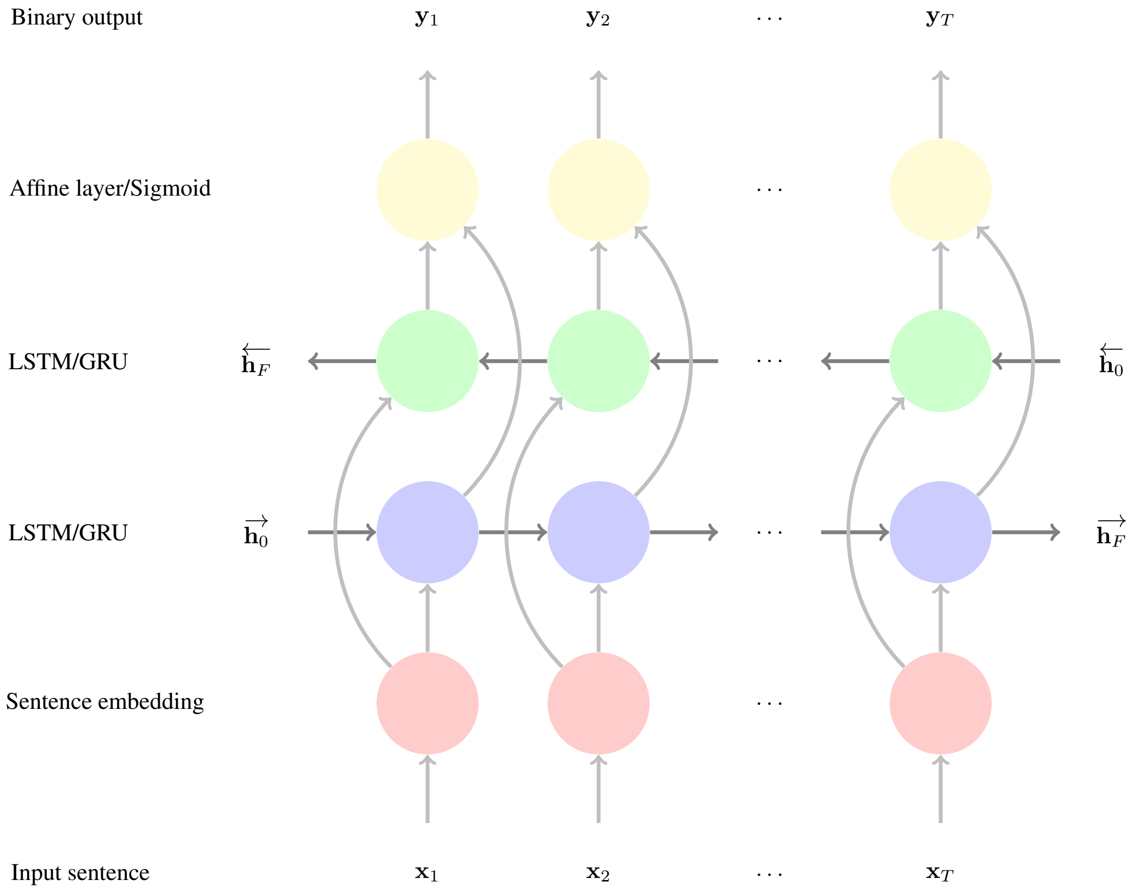


Figure 20: The bidirectional LSTM/GRU recurrent network. At each time step, an output is produced as a function of the previous and following hidden states, and the sentence input.

The second architecture, shown in Figures 21 and 22, is a seq2seq model using an Attention mechanism (see Section 2.3.1). The implementation follows the original mechanism introduced by Bahdanau, Cho, and Bengio (2014), with inspiration for the code implementation taken from Lin et al. (2017): we create the annotations of the input sentence $\mathbf{H} = [\mathbf{h}_1\mathbf{h}_2 \ldots \mathbf{h}_T]$ using the concatenation of the hidden state at both directions at

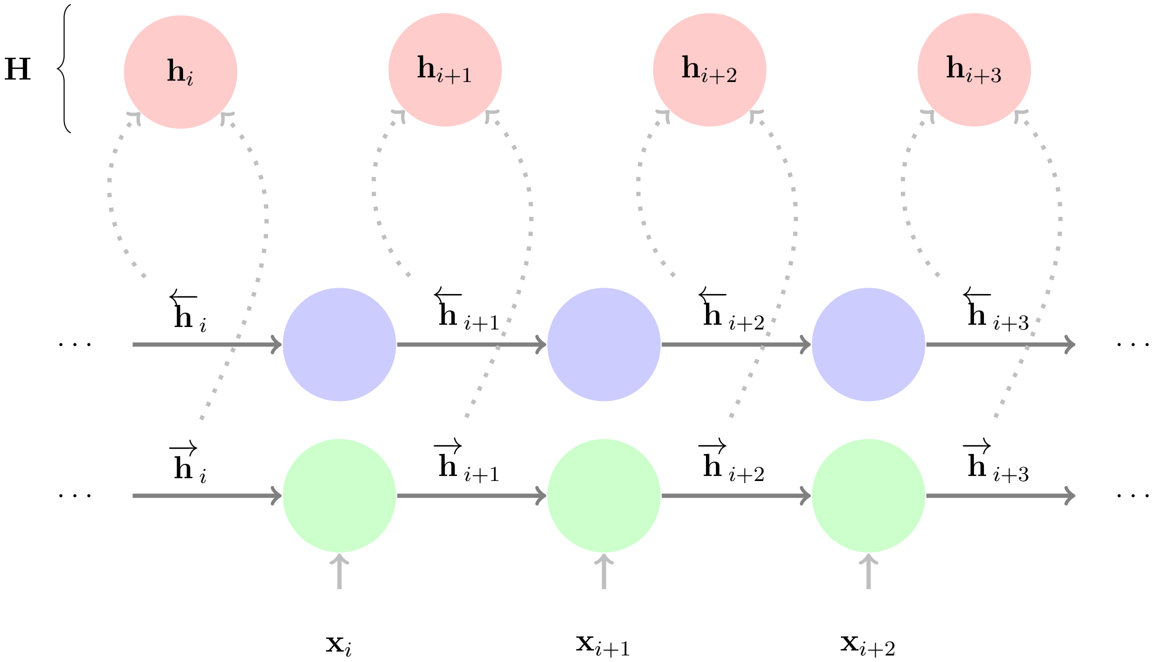

H
$\mathbf{h}_i$
$\mathbf{h}_{i+1}$
$\mathbf{h}_{i+2}$
$\mathbf{h}_{i+3}$
$\overleftarrow{\mathbf{h}}_i$
$\overleftarrow{\mathbf{h}}_{i+1}$
$\overleftarrow{\mathbf{h}}_{i+2}$
$\overleftarrow{\mathbf{h}}_{i+3}$
$\overrightarrow{\mathbf{h}}_i$
$\overrightarrow{\mathbf{h}}_{i+1}$
$\overrightarrow{\mathbf{h}}_{i+2}$
$\overrightarrow{\mathbf{h}}_{i+3}$
...
$\mathbf{x}_i$
$\mathbf{x}_{i+1}$
$\mathbf{x}_{i+2}$

Figure 21: The Attention Encoder. The hidden states at each time step are concatenated to create the annotation of the corresponding input token. These annotations are representations of each of the input tokens, containing information from all other tokens in the input sequence.

each time step, $\mathbf{h}_i = \overrightarrow{\mathbf{h}_i}; \overleftarrow{\mathbf{h}_i}$, as shown in Figure 21. Each annotation is a representation that summarises the current input token along with all other tokens before and after. These annotations will be afterwards weighted at each time step during the decoding phase in order to tell the Decoder which tokens to pay attention to produce the corresponding output.

After the annotations are created, the Decoder, shown in Figure 22, takes a randomly initialised hidden state and iteratively outputs a binary sequence, producing each token as a function of the previous hidden state and the weighted annotations. The *attention weights* assigned to each annotations are produced by a feedforward network that is jointly trained with all other components.

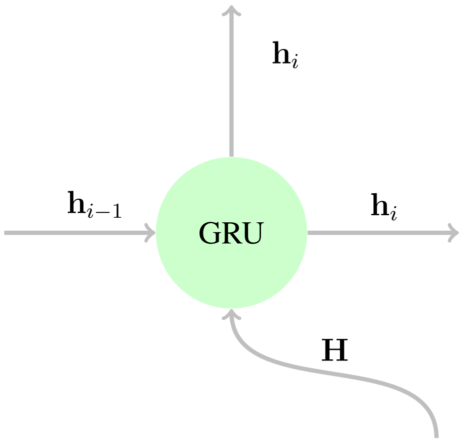


Figure 22: The Attention Decoder. At each time step, the Decoder, implemented as a GRU cell, receives the hidden state from the previous step, $\mathbf{h}_{i-1}$, and the annotations of the input sequence, $\mathbf{H}$. It assigns the corresponding weight to each annotation and uses the result to determine the tokens of the input sequence that best inform the output $\mathbf{h}_i$.

Hyperparameter	**Value**
Batch size	1
Nb epochs	25
Learning rate	1e-3, 4, 5
Weight decay	1e-4
Gradient clip	0.2
Learning rate step	[10
13]	
Learning rate gamma	0.1
Optimiser	ADAM
Bidirectional	Yes
Input size	768

Table 9: Model hyperparameters of the recurrent architectures

5.3 Evaluation of the recurrent architectures

To evaluate the recurrent models, we proceed with a similar approach as with the Sentence Classifier method (see Section 4.2). We compare the output of each instance produced by the recurrent models with the gold standard described in Section 3.3. We trained the models on both the original dataset and the augmented dataset (see Section 3.4) splitting both into training, validation, and testing fractions (75:10:15), using the same split as used for our first approach. As in the case of our first approach, we made sure the testing fraction contained no augmented instances.

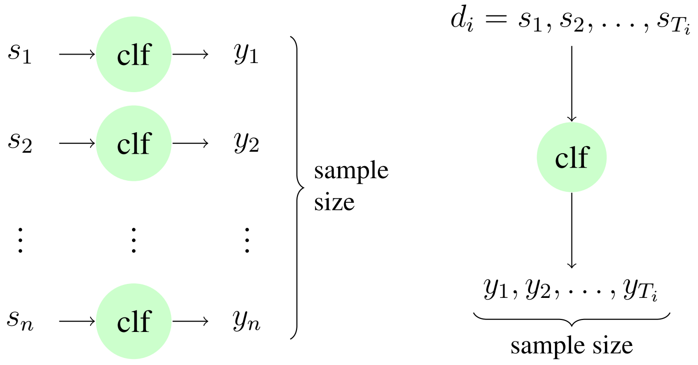
A key difference between the evaluation of this approach and the evaluation of our first approach (see Section 4.2) is that we do not use the batch size as sample size to compute the metrics of Accuracy, Precision, Recall and F_1. Recall that, unlike in the previous approach, the result of a single training iteration is not the binary classification of a single sentence but the output of a binary sequence representing all the sentences in the document at once. Moreover, because of memory constraints, a training iteration using contextual embeddings could only be allocated a batch size of 1. Refer to Figure 23 for a visualisation.

Each entry in the corpus is annotated with a corresponding binary sequence that represents the segmentation of Facts and non-Facts (see Section 3.3); we use these binary sequences to compute the evaluation metrics in our second approach. Results are shown in Table 10.

5.4 Evaluation and discussion of the segmentation task

Using the same heuristic described in Section 4.3, we evaluated the text segmentation task over the test set of annotated cases using the gold standard described in Section 3.3. Results are shown in Tables 11, 12, and 13, and Figures A.1 through A.5 in Appendix A.

As the tables and figures show, the recurrent architectures are able to produce a higher


s_1
clf
y_1
s_2
clf
y_2
s_n
clf
y_n
sample size
$d_i = s_1, s_2, \ldots, s_{T_i}$
clf
$\underbrace{y_1, y_2, \ldots, y_{T_i}}_{\text{sample size}}$

(a) First approach

(b) Second approach

Figure 23: Visual representation of the sample size used to compute performance metrics. In the first approach the batch size n is used as sample size because each sentence s_i is independently classified, regardless of the context in which it is found. In the second approach, the length T_i of each document d_i is used as sample size because each classified sentence y_i is conditioned on all other sentences in the document and each document .

Architecture	**Accuracy**	**Precision**	**Recall**	**F$_1$**
LSTM (original)	0.9865	0.9874	0.9884	0.9879
LSTM (augmented by 2)	0.9638	0.9690	0.9718	0.9704
LSTM (augmented by 3)	0.9661	0.9701	0.9762	0.9731
GRU (original)	0.9888	0.9883	0.9918	0.9900
GRU (augmented by 2)	0.9611	0.9645	0.9719	0.9682
GRU (augmented by 3)	0.9601	0.9674	0.9691	0.9683
Attn (original)	0.9268	0.9123	0.9268	0.8966
Attn (augmented by 2)	0.9529	0.9399	0.9529	0.9279
Attn (augmented by 3)	0.9529	0.9399	0.9529	0.9279

Table 10: Performance of the recurrent models using both the original and the augmented datasets

incidence of documents whose segmentation differs by a single sentence from the gold standard, while, unlike the Sentence Classifier method (see Chapter 4), also producing a much more evenly spread distribution of offsets. For example, Table 11a shows that the highest number of documents segmented precisely at the point indicated by the gold standard (the figure in bold text, 468) corresponds to $\gamma = 1.4$, which corresponds to 70% of the dataset, though the percentage of documents offset by at most 1 sentence $(33+468+16)$

is 78%, which is considerably higher than the performance obtained with the Sentence Classifier method (see Section 4.3.1). Additionally, Figure 24 shows the trend of increasing values of γ corresponding to an increased performance; the greater the value of γ, the greater the number of documents segmented exactly where the gold standard indicated.

The augmented dataset using PLSDA (see Section 3.4), once again, shows no significant improvement in performance, compared to the original dataset. See Figures A.4 and A.5. See Appendix A for additional figures illustrating the performance of the other models.

It is interesting to note that, unlike the Sentence Classifier method, increasing the value of γ leads to an increase in performance, as evidenced in Figures A.4 and A.5, where the fraction of segmented documents whose offset is at most 1 increases as γ increases. This suggests that, despite its better performance in the intrinsic evaluation (see Section 5.3), the recurrent architecture models continue to underestimate the splitting point of a given document. The high intrinsic evaluation also casts additional doubt on the original assumption that *Facts* and *non-Facts* are contiguous and that the former must always precede the latter: similarly to what we observed in Section 4.3.1 high-precision and recall values do not account for the issue of the considerable fraction (17% for $\gamma = 1.4$ and even higher for lower values) of documents whose splitting index is underestimated by more than 4 sentences. This fraction is, nevertheless, a considerable improvement over the one obtained using the Sentence Classifier method (79% for $\gamma = 1.4$), suggesting, ultimately, that the recurrent nature of the models described in this chapter, which account for the semantic dependency of each sentence on its context, are a better approach at segmenting a document into *Facts* and *non-Facts*.

Finally, the reduced performance produced by the use of the PLSDA technique with the LSTM/GRU model suggests that the overall performance of all three models would not significantly increase if there were more data available for training.

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
γ	0.6	518	42	57	21	11	5	2	0	0	1	3
	0.8	326	70	73	90	74	15	1	5	0	0	6
	1	179	8	4	5	46	385	15	4	4	0	10
	1.2	140	2	10	6	45	421	14	6	4	2	10
	1.4	112	2	6	7	50	**440**	17	5	5	1	15

(a) Original dataset

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
γ	0.6	520	42	57	18	12	4	1	2	1	1	2
	0.8	324	73	74	88	73	13	6	1	0	2	6
	1	181	9	6	5	25	397	18	5	4	1	9
	1.2	143	4	8	7	23	430	22	7	5	0	11
	1.4	117	4	5	8	25	**450**	22	8	6	1	14

(b) Dataset augmented by a factor of 2

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
γ	0.6	516	42	58	21	11	5	2	1	0	3	1
	0.8	318	72	76	88	75	12	5	6	1	2	5
	1	179	9	5	7	24	387	25	5	4	1	14
	1.2	137	7	9	8	21	423	28	8	4	1	14
	1.4	111	4	6	11	23	**446**	26	7	5	1	20

(c) Dataset augmented by a factor of 3

Table 11: Sentence offset results for the different values of γ using the LSTM network, and the original and augmented datasets

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
	0.6	517	44	60	23	10	3	1	0	0	0	2
	0.8	319	74	75	89	77	16	1	3	2	1	3
γ	1	176	8	2	6	30	408	16	1	4	0	9
	1.2	137	2	7	7	30	445	17	0	4	1	10
	1.4	112	2	2	9	33	**468**	16	3	3	2	10

(a) Original dataset

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
	0.6	516	43	57	22	12	5	2	0	0	0	3
	0.8	318	73	77	89	75	13	5	3	1	0	6
γ	1	178	9	3	7	24	394	27	3	3	1	11
	1.2	139	4	8	7	22	426	32	6	3	1	12
	1.4	111	4	4	10	22	**448**	33	7	4	1	16

(b) Dataset augmented by a factor of 2

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
	0.6	516	45	57	23	11	5	0	0	1	0	2
	0.8	320	72	79	84	78	15	3	5	0	0	4
γ	1	177	9	2	4	30	400	23	1	4	0	10
	1.2	138	3	7	5	29	434	27	1	4	1	11
	1.4	111	3	4	7	32	**456**	25	3	4	2	13

(c) Dataset augmented by a factor of 3

Table 12: Sentence offset results for the different values of γ using the GRU network, and the original and augmented datasets

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
	0.6	518	42	57	21	11	5	2	0	0	1	3
	0.8	326	70	73	90	74	15	1	5	0	0	6
γ	1	179	8	4	5	46	385	15	4	4	0	10
	1.2	140	2	12	7	50	406	14	8	9	2	10
	1.4	112	2	6	7	66	**413**	17	10	11	1	15

(a) Original dataset

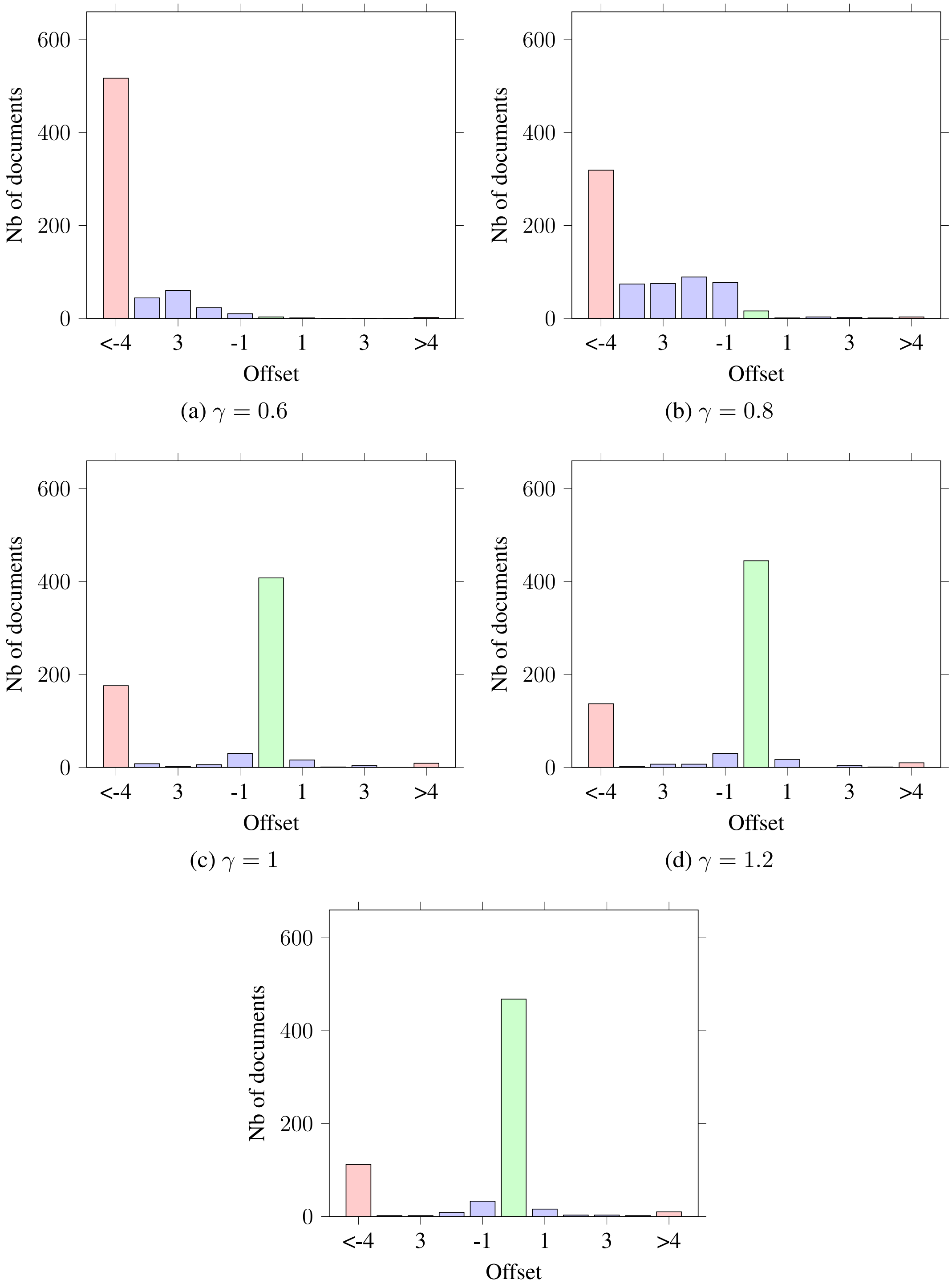
		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
	0.6	524	42	51	18	14	5	2	0	0	0	4
	0.8	326	79	68	88	74	13	1	5	0	0	6
γ	1	179	8	4	5	36	390	13	10	5	0	10
	1.2	140	6	12	9	48	401	13	8	9	4	10
	1.4	112	5	7	9	66	**402**	16	15	12	1	15

(b) Dataset augmented by a factor of 2

		Offset										
		< -4	-4	-3	-2	-1	0	1	2	3	4	> 4
	0.6	518	56	45	20	8	5	1	1	0	1	5
	0.8	326	70	73	90	74	15	1	5	0	0	6
γ	1	179	8	7	5	46	381	16	0	8	0	10
	1.2	145	2	16	7	32	407	16	7	9	7	12
	1.4	108	2	7	9	66	**408**	17	10	11	5	17

(c) Dataset augmented by a factor of 3

Table 13: Sentence offset results for the different values of γ using the seq2seq Attention mechanism, and the original and augmented datasets



(a) $\gamma = 0.6$

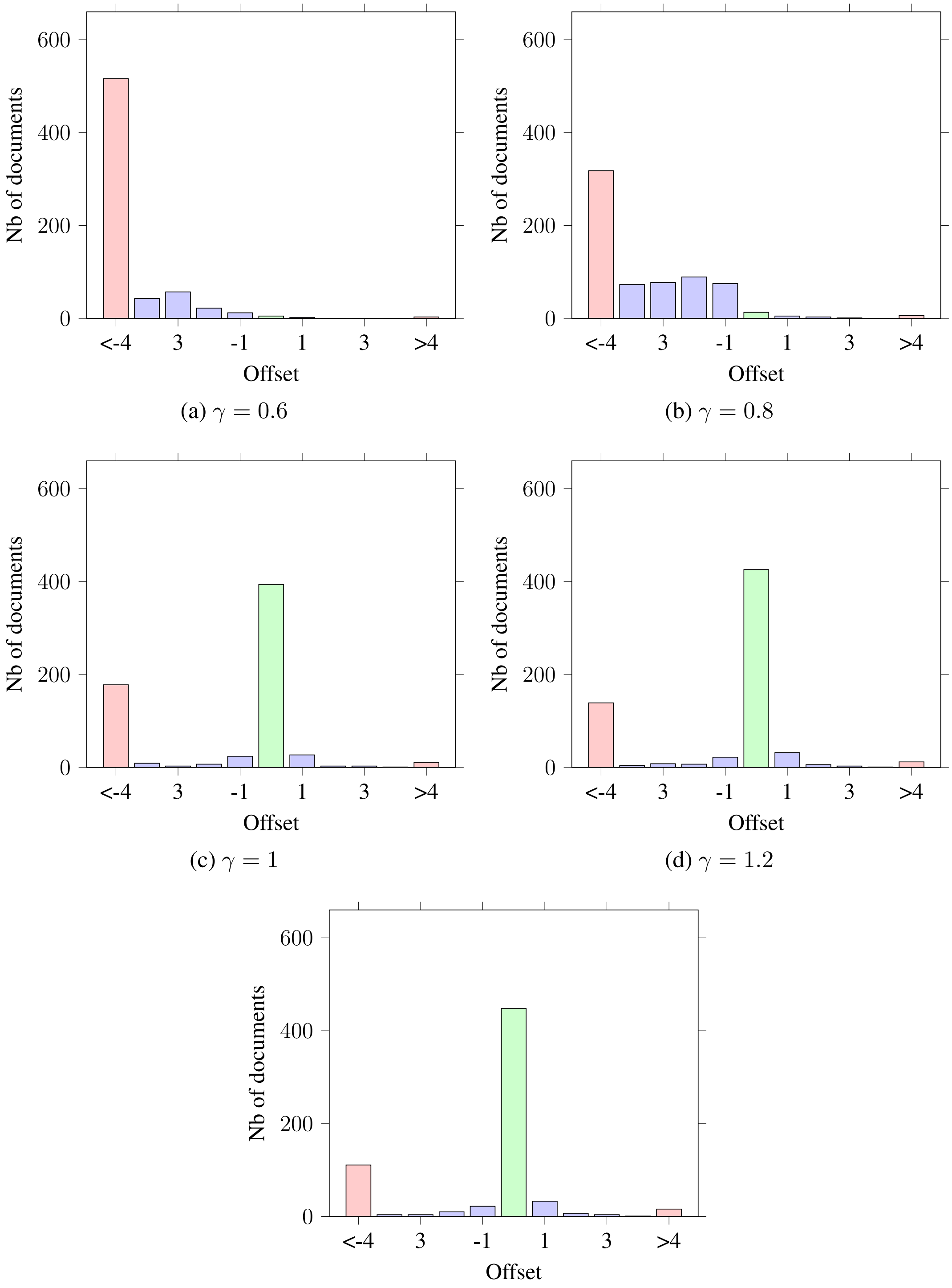
(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure 24: Sentence offset distribution for the different values of γ using a GRU network on the original dataset



(a) $\gamma = 0.6$

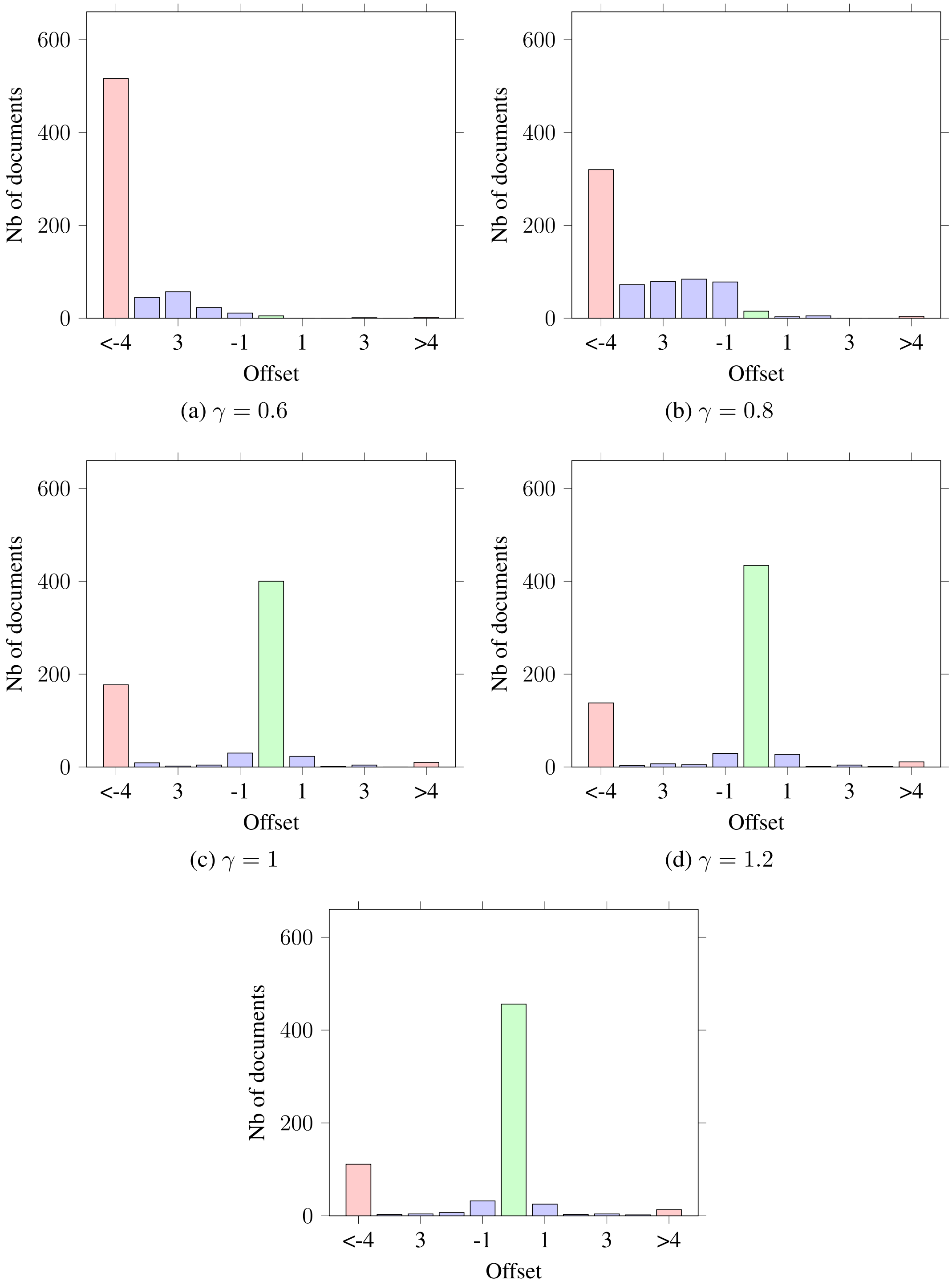
(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure 25: Sentence offset distribution for the different values of γ using a GRU network on the dataset augmented by a factor of 2



(a) $\gamma = 0.6$

(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure 26: Sentence offset distribution for the different values of γ using a GRU network on the dataset augmented by a factor of 3

Chapter 6

Summary and future work

In this chapter, we provide a summary of the approaches used in this thesis to perform Discourse Segmentation of legal documents in order to extract case-relevant *Facts*. We then provide avenues for future work.

6.1 Summary of the thesis

The goal of the thesis was to extract the case relevant facts from texts of judicial rulings by means of neural approaches. To achieve this goal we developed two methods based on Text Segmentation. Both methods rely on the stated assumption from Figure 2, that facts will be listed and stated before the analysis that produces a decision based on the facts. This thesis was developed as part of the process of building an extrinsic evaluation of the JusticeBot project, developed by the CyberJustice laboratory and the Université de Montréal.

Our first method (see Chapter 4) is based on the independent classification of each sentence in a document and the subsequent segmentation of the document based on the metrics of length and purity. The second method (see Chapter 5) is based on a recurrent architecture where sentence classification is done contextually, and the subsequent segmentation is also based on length and purity. Both models were trained and tested using an annotated corpus derived from a dataset of legal cases concerning landlord-tenant disputes using the original corpus and an augmented corpus produced with PLSDA.

The first method, the Sentence Classifier method produced the following highlights as part of the intrinsic evaluation: the classifier trained on the original dataset produced an accuracy of 0.79 and an F_1 measure of 0.77 for the class *Fact*; the classifiers trained on the augmented datasets performed similarly. As part of the extrinsic evaluation, using the original dataset, the splitting index value that yielded the greatest percentage of sentences offset by fewer than 1 with respect to the gold standard, for all models (base and augmented by 2 and 3) was $\gamma = 1.4$.

The second method, the Recurrent Architecture (see Chapter 5), consisted of 3 distinct recurrent models: an LSTM network, a GRU network, and an Attention Encoder-Decoder. This method produced the following highlights as part of the intrinsic evaluation: overall, the GRU model produced the best performance across the board, with an accuracy of 0.99 and an F_1 score of 0.99 on the original dataset, and an accuracy of 0.95 and an F_1 score of 0.93 on the augmented dataset. The extrinsic evaluations (see Section 5.4) produced a smaller percentage of documents whose segmentation point fell short of the point indicated by the gold standard, compared to the Sentence Classifier method, while also considerably increasing the number of documents whose offset from the predicted segmentation point was fewer than 1 sentence.

The second method produced better extrinsic evaluation results than the first, increasing the number of documents whose offset from the gold standard was less than 1 sentence from 5% to 78% percent for their best value of the splitting index γ.

The PLSDA technique of data augmentation (see Section 3.4) produced no significant improvements in performance on either methods, which suggests that using more real-world data to train our models might not be particularly important to increase performance.

6.2 Future work

Following our work, it would be interesting to investigate the following research questions:

1. Our work was based on the stated assumption in Figure 2, which was in turn based on the IRAC legal writing model (Beazley 2018). Future work should explore the validation of the assumption, or proposing additional axioms based on more refined models of writing. An implicit assumption we made was that documents from the legal system followed the rule in the province of Québec, which follows both Civil Law and Common Law, instead of solely Common Law like the rest of Canada. It might be the case that these two distinct legal traditions follow different discourse conventions.

2. Both our methods showed no significant improvement of performance with the augmented dataset. Future work should focus on better text mining techniques in order to obtain a larger training corpus, in order to concretely demonstrate that additional data might not be enough to increase the performance of our models, in particular, the Sentence Classifier.

3. Additionally, future work might also make exclusive use of contextual word and sentence embeddings, such as the ones produced by BERT and its derivatives, which have consistently shown to outperform non-contextual word embeddings in several benchmarks that use them (Devlin et al. 2018). The Sentence Classifier method could be fitted with BERT or a BERT-like component in order to use contextual word embeddings and so take advantage of camemBERT's fine-tuning capabilities; it could also be fitted with an additional encoding module in the form of a camemBERT module with a feedforward linear head in order to enrich the final representation.

4. Finally, more work needs to be done in terms of model size optimisation in order

to incorporate camemBERT's fine-tuning capabilities into the Recurrent Architecture approach. As it stands, the long sequence of sentences typical of any training instance impose a computational challenge that makes it impossible to train camemBERT with our current downstream task. Perhaps techniques such as knowledge distillation (Hinton, Vinyals, and Dean 2015) combined with BERT like models, such as DistilBERT (Sanh et al. 2019).

Bibliography

Allan, James et al. (May 2003). “Topic Detection and Tracking Pilot Study Final Report”. In: DOI: 10.1184/R1/6610943.v1. URL: https://kilthub.cmu.edu/articles/Topic_Detection_and_Tracking_Pilot_Study_Final_Report/6610943.

Attardi, Giuseppe (June 2006). “Experiments with a multilanguage non-projective dependency parser”. In: *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*. New York City, pp. 166–170.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (May 2014). “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473*.

Baroni, Marco et al. (Feb. 2009). “The WaCky wide web: a collection of very large linguistically processed web-crawled corpora”. In: *Language Resources and Evaluation* 43.3, pp. 209–226.

Beazley, Mary Beth (2018). *A practical guide to appellate advocacy*. Aspen Publishers, pp. 95–96.

Beeferman, Doug, Adam Berger, and John Lafferty (Feb. 1999). “Statistical models for text segmentation”. In: *Machine Learning* 34.1-3, pp. 177–210.

Bermingham, Adam and Alan F Smeaton (July 2009). “A study of inter-annotator agreement for opinion retrieval”. In: *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM. Boston, MA, pp. 784–785.

Blei, David M, Andrew Y Ng, and Michael I Jordan (Jan. 2003). “Latent Dirichlet Allocation”. In: *Journal of Machine Learning Research* 3.Jan, pp. 993–1022.

Campbell, Samuel et al. (Mar. 2018). *ProceZeus: An AI-powered chatbot that aims to facilitate access to judicial proceedings involving rental housing issues in Québec*. Retrieved on 2021-06-10. Version 1.0. URL: https://github.com/Cyberjusticelab/JusticeAI.

Cho, Kyunghyun et al. (Sept. 2014). “Learning phrase representations using RNN encoder-decoder for statistical machine translation”. In: *arXiv preprint arXiv:1406.1078*.

Chung, Junyoung et al. (2014). “Empirical evaluation of gated recurrent neural networks on sequence modeling”. In: *arXiv preprint arXiv:1412.3555*.

Cortes, Corinna and Vladimir Vapnik (1995). “Support vector machine”. In: *Machine learning* 20.3, pp. 273–297.

Dell’Orletta, Felice et al. (May 2012). “The SPLeT-2012 shared task on dependency parsing of legal texts”. In: *Proceedings of the 4th Workshop on Semantic Processing of Legal Texts*. Istanbul, pp. 42–51.

Devlin, Jacob et al. (2018). “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805*.

Dragoni, Mauro et al. (2015). “Combining natural language processing approaches for rule extraction from legal documents”. In: *AI Approaches to the Complexity of Legal Systems*. Braga, Portugal: Springer, pp. 287–300.

Edmonds, Philip (2002). “SENSEVAL: The evaluation of word sense disambiguation systems”. In: *ELRA newsletter* 7.3, pp. 5–14.

Felsenburg, Miriam E and Laura P Graham (2010). “Beginning legal writers in their own words: Why the first weeks of legal writing are so tough and what we can do about it”. In: *Journal of the Legal Writing Institute* 16, pp. 223–312.

Fix, Evelyn and JL Hodges (1952). “Discriminatory analysis: Nonparametric discrimination: Consistency properties”. In: *USAF School of Aviation Medicine, Project*, pp. 21–49.

Flanigan, Emily J (2019). *A Lawyer Writes: A Practical Guide to Legal Analysis*. Chicago, IL: American Association of Law Libraries, pp. 81–83.

Garner, Bryan A (2009). “Garner on language and writing”. In: American Bar Association, pp. 402–422.

Grabmair, Matthias et al. (June 2015). “Introducing LUIMA: An experiment in legal conceptual retrieval of vaccine injury decisions using a UIMA-type system and tools”. In: *Proceedings of the 15th International Conference on Artificial Intelligence and Law*. ACM. San Diego, CA, pp. 69–78.

Greff, Klaus et al. (2016). “LSTM: A Search Space Odyssey”. In: *IEEE transactions on neural networks and learning systems* 28.10, pp. 2222–2232.

Hage, Jaap, Antonia Waltermann, and Bram Akkermans (2017). *Introduction to law*. Springer, pp. 22–23.

Harris, Zellig S (1954). “Distributional structure”. In: *Word* 10.2-3, pp. 146–162.

Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean (2015). “Distilling the Knowledge in a Neural Network”. In: arXiv: `1503.02531 [stat.ML]`.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long short-term memory”. In: *Neural computation* 9.8, pp. 1735–1780.

Honnibal, Matthew et al. (2020). *spaCy: Industrial-strength Natural Language Processing in Python*. DOI: 10.5281/zenodo.1212303. URL: https://doi.org/10.5281/zenodo.1212303.

Kim, Yoon (Sept. 2014). “Convolutional neural networks for sentence classification”. In: *arXiv preprint arXiv:1408.5882*.

Lai, Siwei et al. (Feb. 2015). “Recurrent convolutional neural networks for text classification”. In: *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pp. 2267–2273.

Law.com, ed. (2020). *Fact*. URL: https://dictionary.law.com/Default.aspx?selected=718 (visited on 06/15/2020).

Lin, Zhouhan et al. (2017). “A structured self-attentive sentence embedding”. In: *arXiv preprint arXiv:1703.03130*.

Liu, Yinhan et al. (2019). “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692*.

Maat, Emile de and Radboud Winkels (June 2009). “A next step towards automated modelling of sources of law”. In: *Proceedings of the 12th International Conference on Artificial Intelligence and Law*. ACM, pp. 31–39.

Maat, Emile de, Radboud Winkels, and Tom Van Engers (Dec. 2006). “Automated detection of reference structures in law”. In: *Frontiers in Artificial Intelligence and Applications*, p. 41.

Martin, Louis et al. (Nov. 2019). “CamemBERT: a Tasty French Language Model”. In: *arXiv e-prints*, arXiv:1911.03894, arXiv:1911.03894. arXiv: 1911.03894 [cs.CL].

Merriam-Webster, ed. (2020). *Fact*. URL: https://www.merriam-webster.com/dictionary/fact#legalDictionary (visited on 06/15/2020).

Mikolov, Tomas et al. (2013). “Efficient Estimation of Word Representations in Vector Space”. In: arXiv: 1301.3781 [cs.CL].

Miller, George A (1995). “WordNet: a lexical database for English”. In: *Communications of the ACM* 38.11, pp. 39–41.

Misra, Hemant et al. (Nov. 2009). “Text segmentation via topic modeling: an analytical study”. In: *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, pp. 1553–1556.

Mozetič, Igor, Miha Grčar, and Jasmina Smailović (May 2016). “Multilingual Twitter sentiment classification: The role of human annotators”. In: *PloS One* 11.5, e0155036.

Nance, Dale A (1987). “The Best Evidence Principle”. In: *Iowa Law Review* 73, p. 227.

Olah, Chris (Aug. 2015). *Understanding LSTM Networks*. Retrieved on 2020-06-16. `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

Pang, Bo, Lillian Lee, et al. (July 2008). "Opinion mining and sentiment analysis". In: *Foundations and Trends in Information Retrieval* 2.1–2, pp. 1–135.

Peters, Matthew E et al. (2018). "Deep contextualized word representations". In: *arXiv preprint arXiv:1802.05365*.

Ponte, Jay M and W Bruce Croft (1997). "Text segmentation by topic". In: *International Conference on Theory and Practice of Digital Libraries*. Springer, pp. 113–125.

Rush, Alexander, Vincent Nguyen, and Guillaume Klein (Apr. 2018). *The Annotated Transformer*. Retrieved on 2020-06-16. `http://nlp.seas.harvard.edu/2018/04/03/attention.html`.

Sagae, Kenji and Jun-Ichi Tsujii (2010). "Dependency parsing and domain adaptation with data-driven LR models and parser ensembles". In: *Trends in Parsing Technology*. Springer, pp. 57–68.

Salaün, Olivier et al. (2020). "Analysis and Multilabel Classification of Quebec Court Decisions in the Domain of Housing Law". In: *International Conference on Applications of Natural Language to Information Systems*. Springer, pp. 135–143.

Sanh, Victor et al. (2019). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: arXiv: `1910.01108 [cs.CL]`.

Savelka, Jaromír and Kevin D Ashley (2017). "Using conditional random fields to detect different functional types of content in decisions of united states courts with example application to sentence boundary detection". In: *Workshop on Automated Semantic Analysis of Information in Legal Texts*, p. 10.

— (2018). "Segmenting US Court Decisions into Functional and Issue Specific Parts." In: *JURIX: The 31st International Conference on Legal Knowledge and Information Systems*, pp. 111–120.

Sennrich, Rico, Barry Haddow, and Alexandra Birch (2015). "Neural machine translation of rare words with subword units". In: *arXiv preprint arXiv:1508.07909*.

Stein, Alex (2005). "Foundations of evidence law". In: Oxford University Press. Chap. Epistemological Corollary.

Strapparava, Carlo and Rada Mihalcea (2007). "Semeval-2007 task 14: Affective text". In: *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pp. 70–74.

Suárez, Pedro Javier Ortiz, Benoît Sagot, and Laurent Romary (2019). "Asynchronous Pipeline for Processing Huge Corpora on Medium to Low Resource Infrastructures". In: *Challenges in the Management of Large Corpora (CMLC-7) 2019*, p. 9.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (Dec. 2014). "Sequence to sequence learning with neural networks". In: *Advances in Neural Information Processing Systems*. Montréal, QC, pp. 3104–3112.

Taylor, Wilson L (1953). ""Cloze procedure": A new tool for measuring readability". In: *Journalism quarterly* 30.4, pp. 415–433.

Utiyama, Masao and Hitoshi Isahara (July 2001). "A statistical model for domain-independent text segmentation". In: *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*. Toulouse, France, pp. 499–506.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems*. Long Beach, CA, pp. 5998–6008.

Wald, Patricia M (Oct. 1995). "The rhetoric of results and the results of rhetoric: Judicial writings". In: *The University of Chicago Law Review* 62.4, pp. 1371–1419.

Walker, Vern R, Ji Hae Han, et al. (2017). "Semantic types for computational legal reasoning: propositional connectives and sentence roles in the veterans' claims dataset". In: *Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law*. London, UK, pp. 217–226.

Walker, Vern R, Bernadette C Lopez, et al. (2015). "Representing the Logic of Statutory Rules in the United States". In: *Logic in the Theory and Practice of Lawmaking*. Springer, pp. 357–381.

Westermann, Hannes et al. (June 2019). "Using Factors to Predict and Analyze Landlord-Tenant Decisions to Increase Access to Justice". In: *Proceedings of the 17th International Conference on Artificial Intelligence and Law*. ACM. Montréal, QC, pp. 133–142.

Wolf, Thomas et al. (2019). "HuggingFace's Transformers: State-of-the-art Natural Language Processing". In: *ArXiv* abs/1910.03771.

Xiang, Rong et al. (2020). "Lexical Data Augmentation for Text Classification in Deep Learning". In: *Canadian Conference on Artificial Intelligence*. Springer, pp. 521–527.

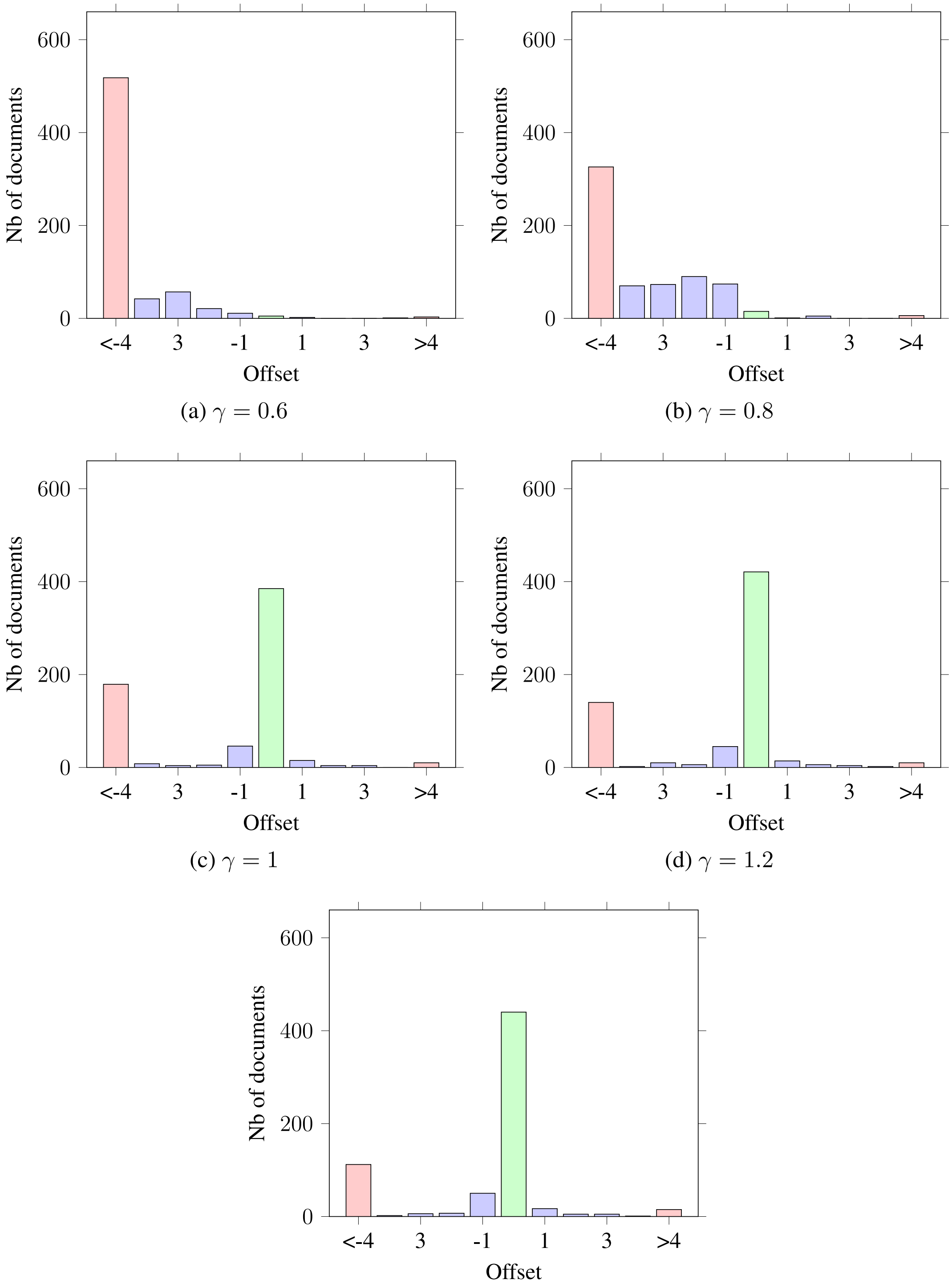
Zhang, Xiang, Junbo Zhao, and Yann LeCun (Dec. 2015). "Character-level convolutional networks for text classification". In: *Advances in Neural Information Processing Systems*. Montréal, QC, pp. 649–657.

Zhou, Peng et al. (Apr. 2016). "Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling". In: *arXiv preprint arXiv:1611.06639*.

Appendix A

Additional bar plots of the segmentation task using Recurrent Architectures

Here we present a set of additional bar plots that illustrate the number of sentences offset by a varying number of sentences from the predicted segmentation point given a fixed value of γ. For example, consider subfigure (e) from Figure A.1, given $\gamma = 1.4$, we can see that around 2,000 documents are underestimated in terms of segmentation; around 1,000 sentences are segmented at the exact point where the gold standard indicates the segmentation, and finally a similar figure of a little over 1,000 thousand documents are overestimated and segmented more than 4 sentences beyond what the gold standard expected.



(a) $\gamma = 0.6$

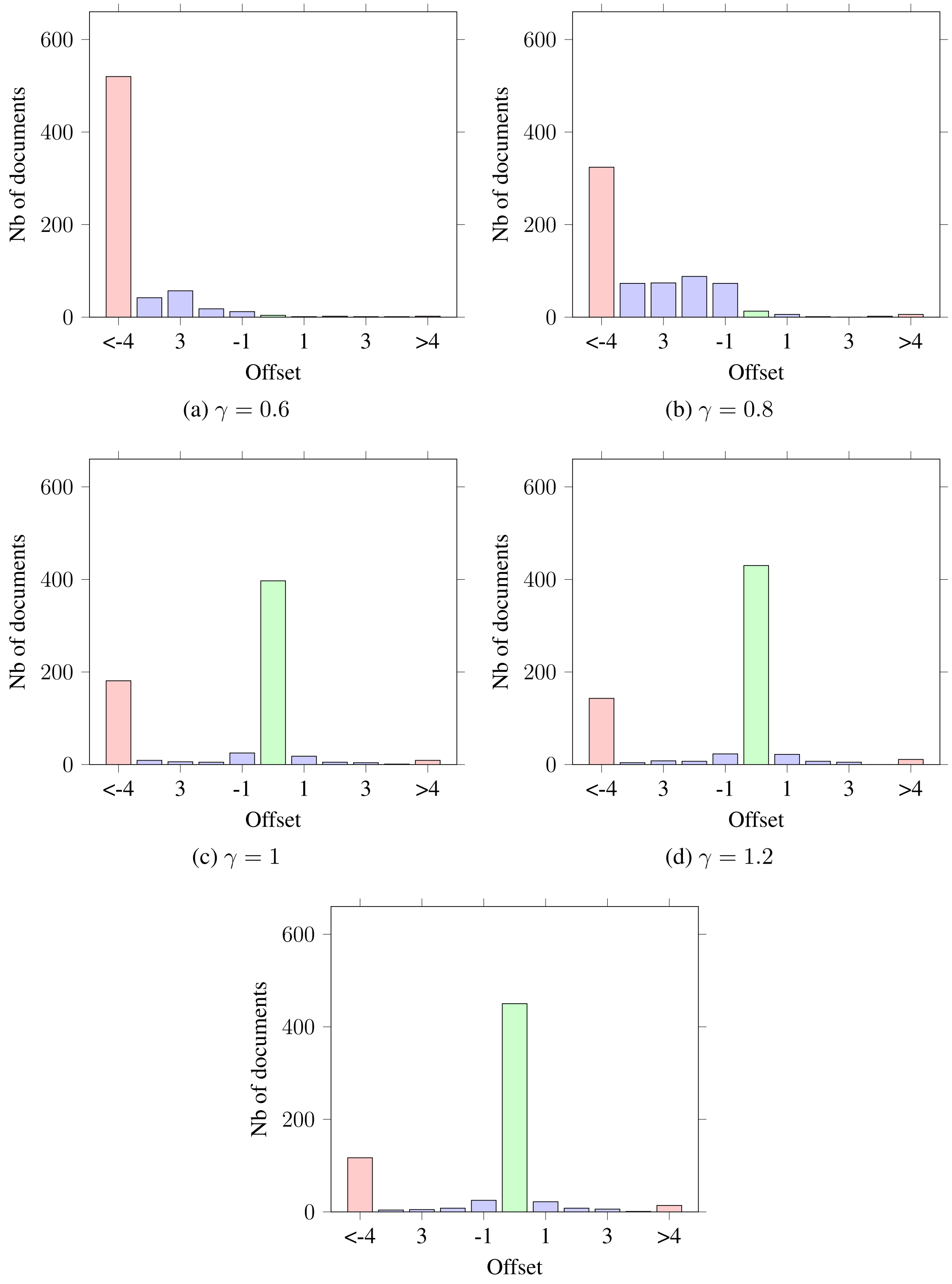
(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure A.1: Sentence offset distribution for the different values of γ using an LSTM network on the original dataset



(a) $\gamma = 0.6$

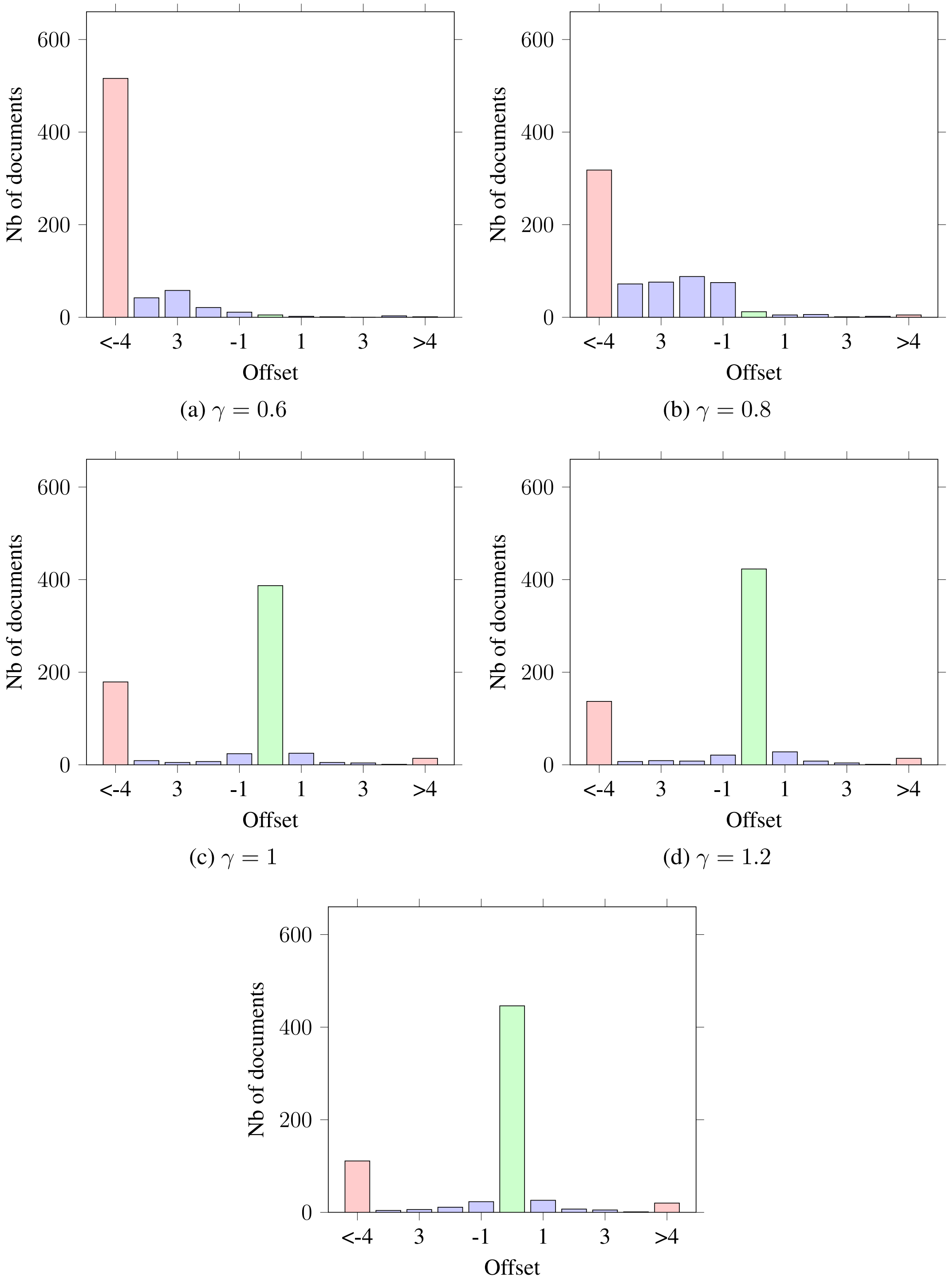
(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure A.2: Sentence offset distribution for the different values of γ using an LSTM network on the dataset augmented by a factor of 2



(a) $\gamma = 0.6$

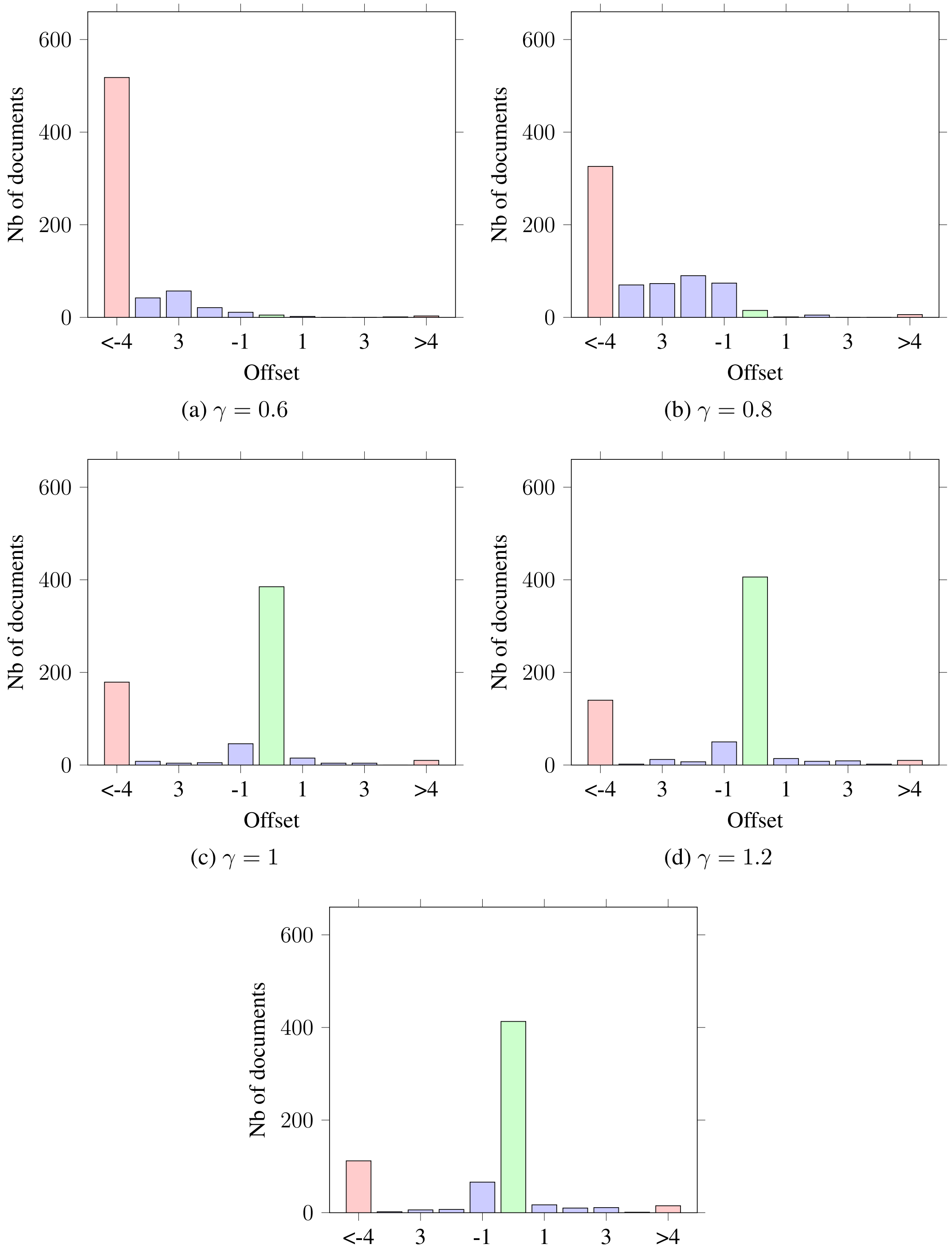
(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure A.3: Sentence offset distribution for the different values of γ using an LSTM network on the dataset augmented by a factor of 3



(a) $\gamma = 0.6$

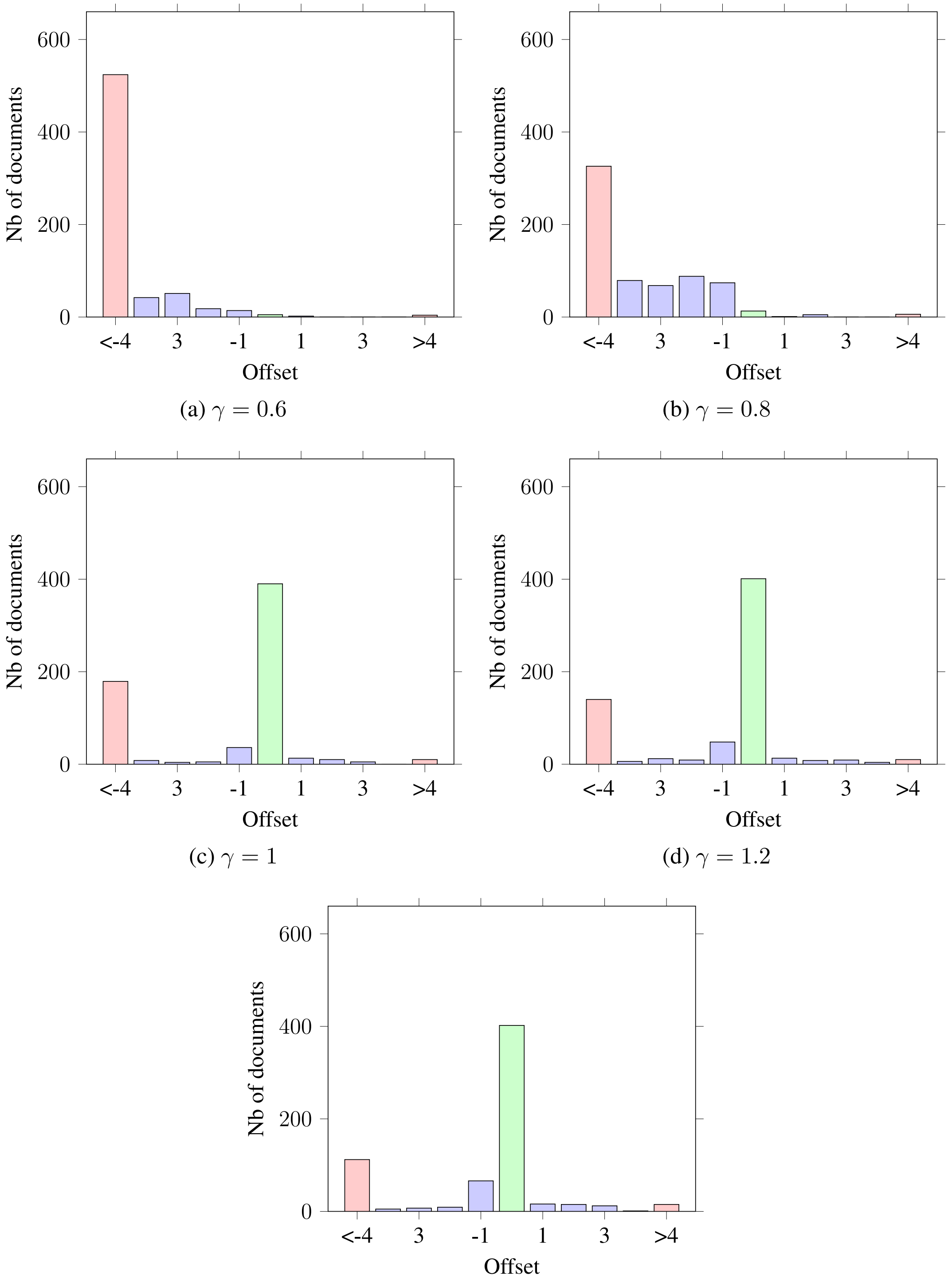
(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure A.4: Sentence offset distribution for the different values of γ using the seq2seq Attention mechanism on the original dataset



(a) $\gamma = 0.6$

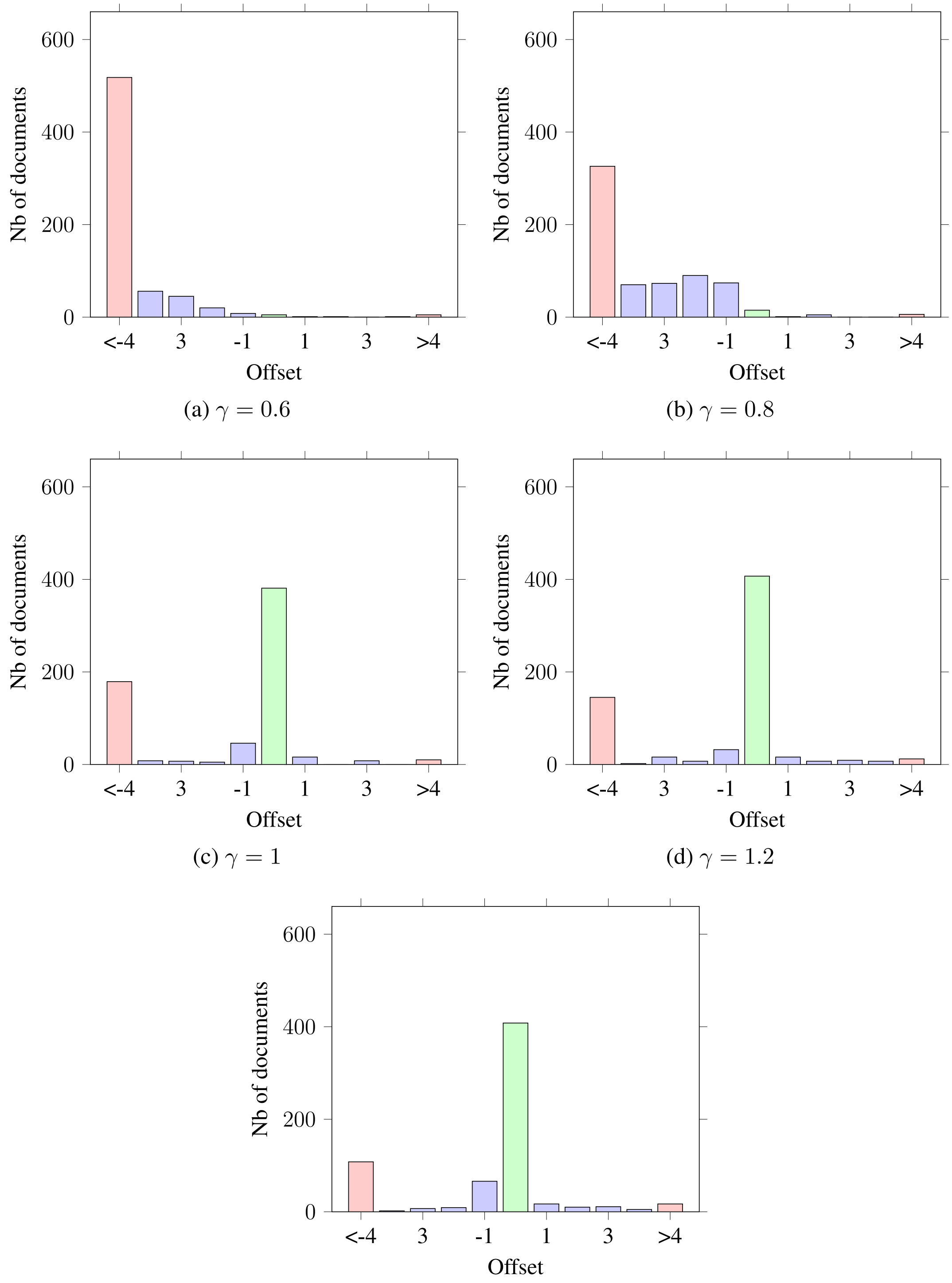
(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure A.5: Sentence offset distribution for the different values of γ using the seq2seq Attention mechanism on the dataset augmented by a factor of 2



(a) $\gamma = 0.6$

(b) $\gamma = 0.8$

(c) $\gamma = 1$

(d) $\gamma = 1.2$

(e) $\gamma = 1.4$

Figure A.6: Sentence offset distribution for the different values of γ using the seq2seq Attention mechanism on the dataset augmented by a factor of 3