POLITECNICO
MILANO 1863

Master of Science in Computer Science and Engineering
School of Industrial and Information Engineering

BASILISCo: an advanced methodology for text complexity calculation

Supervisor: Prof. Licia SBATTELLA
Co-supervisor: Ing. Roberto TEDESCO

Master Thesis of:
Andrea PEREGO
Matr. 898656

Academic Year 2019 - 2020

Politecnico di Milano:
www.polimi.it

School of Industrial and Information Engineering:
www.ingindinf.polimi.it

Abstract

This thesis introduces a novel strategy targeted at tackling the problem of reading complexity by presenting an approach based on the analysis of Lexicon and Semantic. Contrary to the state of the art methods, that proposes a general classification of the reading complexity, this approach generates its strength by the independent analysis of the two mentioned domains. Thanks to this approach, it is possible to provide to a content creator, interested in evaluating the complexity of his work, a more specific analysis of the document by clearly distinguishing the complexity of different areas. This will prove out to be of great benefit for the author, since he will be able to properly adjust the complexity of his work, according to the results provided by the software.

The peculiarity of this approach and the intrinsic innovation introduced is correlated with the modality used to compute the two mentioned complexity.

Lexical Complexity has been implemented using a technique borrowed by a similar task of Natural Language Processing: content selection. The two activities present similar needs, within the content selection task, we need to recognize the concept that best distinguishes a document, meanwhile, in the assessment of lexical complexity, we want to identify which words better discriminate specific levels of complexity.

Syntactic Complexity, instead, has been implemented using a deep learning-based approach. In this case, the difficulty of the task mandated such a choice. While it can be "simple" to associate a word to a specific level of complexity, it is not so easy with grammatical features, unless specific linguistic researches are applied. Given these premises, the choice of using a system that can automatically infer the set of features that characterize each level of complexity is almost mandatory.

The system has been implemented for English, however, it can be easily adapted to other languages, by simply changing the cores corpora. The entire process is, in fact, language independent and can be easily transposed to any other language, for which feasible corpora do exist. This implies that the approach can also be applied in the context of a Second Language Learning (L2 Learning).

Sommario

Questa tesi presenta una nuova strategia mirata ad affrontare il problema della complessità di lettura, presentando un approccio basato sull'analisi del Lessico e della Semantica. Contrariamente ai metodi comuni oggigiorno, che propongono una classificazione generale della complessità di lettura, quest'approccio genera la sua forza dall'analisi indipendente dei due domini menzionati in precedenza. Grazie a quest'approccio, è possibile fornire al creatore di contenuti, interessato nel valutare la complessità del suo lavoro, un'analisi più specifica dell'opera distinguendo chiaramente tra le varie tipologie di complessità. Questo si rivelerà essere un grande beneficio per l'autore, il quale sarà in grado di sistemare la complessità del suo lavoro conformemente al risultato fornito dall'applicativo.

La peculiarità di questo approccio, e di conseguenza la sua innovatività, è associata alla modalità in cui le due complessità sono calcolate.

La Complessità Lessicale è stata implementata usando una tecnica presa in prestito da un compito simile tipico dell'Elaborazione del Linguaggio Naturale (ELN o NLP in inglese): selezione del contenuto. Le due attività presentano dei bisogni simili; nel caso sella selezione di contenuto, vogliamo riconoscere il concetto che meglio distingue un certo documento, mentre, nell'individuazione della complessità lessicale, l'obiettivo è individuare quali parole meglio rappresentano un certo livello di complessità.

La Complessità Sintattica, invece, è stata implementata usando un approccio basato sull'Apprendimento Profondo (o Deep Learning in inglese). La difficoltà del compito ha reso questa scelta quasi obbligatoria. Infatti, mentre può essere "semplice" assegnare una parola ad un certo livello di complessità, non è così semplice con le caratteristiche grammaticali, a meno che non vengano eseguite delle ricerche linguistiche mirate. Data questa premessa, la scelta di usare un sistema in grado di inferire automaticamente l'insieme di elementi che caratterizzano ogni livello di complessità, è quasi obbligatoria.

La procedura è stata implementata per la lingua inglese, tuttavia, può essere facilmente adottata anche ad altri linguaggi, semplicemente cambiando il dataset usato. L'intero processo è infatti indipendente dal linguaggio e può essere facilmente trasposto ad ogni altro linguaggio per cui sono disponibili dei corpora. Questo

implica che l'approccio può essere usato anche in un contesto di apprendimento di una seconda lingua (L2 Learning).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivations

In our current society content creator face a complex struggle when they try to create a text for a specific level of understanding of a language and is not so uncommon to have reader complaining about the effective complexity of a passage or content either because it is too simple compared to their level of knowledge or because it is too complex, leading in both ways to a frustration of the final user.

To access this need, various companies and researchers worked to create classification indexes that can define the proper level of a text compared to a standardized scale. This lead to the definition of common specification such as the Common Score Standard in the US, the Common European Framework of Reference for Languages (CEFR) in Europe and Hanyu Shuiping Kaoshi (HSK) for China, either used for assessing the proficiency of second language learners or for defining the level of knowledge that an individual must possess at a specific scholastic level.

In all these classification systems, however, almost no distinction is poised among the nature of the features considered. The values retrieved, provide a general Reading Complexity level, without specifying if it has to be associated with the lexicon, syntactic, or the semantic used and eventually with which relevance to each of them.

Some researchers tried to assess this deficit by implementing systems with pre-engineered features, each one representing a specific aspect of a document. Unfortunately, due to the intrinsic nature of this approach, the systems turn out to be not transferable to languages different from the one for which they are developed. Furthermore, the high amount of specialist features used will generate confusion in the content creator.

For these reasons, in this thesis, we are going to propose BASILISCo (Bivariate Advanced Syntax Index and Lexical Index for Subsuming Complexity), a system

that tries to satisfy this need, albeit only partially, by providing a diverse and independent complexity classification for both lexicon and syntactic.

1.2 Aims

This research tries to assess the mentioned need by creating a mixed approach that can produce as the result two values: one representing the Syntactic Complexity and the other, the Lexical Complexity of the text. This decision is motivated by the belief that when giving feedback to the content creator is important to provide enough information, allowing him to understand which part is not reflecting the desired difficulty both in term of grammatical structure and words used.

The choice for these two metrics is mainly based on the assumption that Reading Complexity can be sub-categorized in Lexical, Syntactic, and Semantic Complexity, following the main classification of the features used in the recent researches. Among such features, Semantic is strictly dependent by the reader, considering our objective to present a standardized analysis, we are going to ignore it while focusing only on the former two.

In particular, Lexical Complexity associates with the idea that each individual learns words by starting with the simplest ones and subsequently use such terms as building blocks to increase the depth of his vocabulary. Syntactic Complexity, instead, derives from the belief that more complex grammatical structures reflex a deeper knowledge and master of the language. Both notions are intuitive and perfectly reflect the system used to teach a language in the school environment.

1.3 Outline

The present document is subdivided into chapters, each devoted to a fundamental topic required for the comprehension of the thesis.

Chapter 2 briefly presents the story of reading complexity up to the most recent development, posing particular attention to the analysis of Lexical and Syntactic Features. This chapter will also provide background knowledge, so to support the reader in understanding the content of this thesis.

Chapter 3 presents the rationale behind the work, highlighting that even if the thesis evolves in two different paths, a common ideal connects them. This chapter also introduces the datasets that will be used in the research, together with a series of analyses on their nature.

Chapter 4 is responsible for the lexical component of the research, and as such, constitutes one of the two core chapters of this thesis. This chapter, in

particular, will present the method used to compute the Lexical Complexity and the experimental results obtained by implementing such an algorithm. The model will then test the results against low-level metrics identified in previous researches through correlation analysis. Lastly, we will propose a possible scaling system to give semantics to the score obtained from new documents.

Chapter 5 is responsible for the syntactic component of the research, and together with the previous chapter, it is one of the two core chapters of this work. Given the common rationale of the work, this chapter will be structured in a similar way to the previous one, the only difference being that two different models will be proposed and analyzed.

Chapter 6 draws the main conclusions about the thesis work and suggests further developments in this direction.

Chapter 2

Background

2.1 Introduction

One of the most relevant questions in reading research in the past years has been to estimate the difficulty level of a document. This need was born from the will to help students that are practicing to achieving the mastery of a language, either as a step block needed to communicate within the society or as a personal desire.

It is common knowledge that when learning a language, the most useful tools are entertainment contents, mainly including books, tv-series, movies, music, and videogames. For the scope of this research, we will focus only on literature's works.

It is not uncommon for a language teacher to assign books to their students as training material, however, not all books are alike, and some present a complexity higher than others. The teacher needs to be able to correctly estimate the complexity of the book and associate it with the level of the learner.

This task is the responsibility of the teacher since he follows the students during every phase of the learning process and has a better understanding of the level of the individual. Unfortunately, given the high volume of literature works present in the market and the huge difference in the literary inclination of the students, a means to automatically estimate the complexity of publications on a defined scale is needed.

The first studies executed on the matter focused only on superficial characteristics of the text (such as word or sentence length), giving birth to a series of language-dependent formulas labeled as *readability formulas*. Most of these formulas, in an upgraded version, are still used today by companies even if many researchers raised objections against them through the years [7, 8].

Further studies, achieved through the use of Machine Learning approaches revealed that more complex features covering all linguistic aspects of the text can

provide a better result.

In this chapter, we are firstly going to describe the most relevant features in text analysis and the approaches in which they are used. Then we will discuss the more recent and innovative strategies based on machine learning and deep learning, including the required knowledge to understand them.

2.2 Features

When thinking about the complexity of a text and trying to find out which are the most evident linguistic features associated with it, usually, the first one that comes to mind is either the difficulty of the words used or the complexity of the grammatical structures of the sentence. However, these are just the tip of the immense amount of linguistic features that can be taken into consideration when verifying the complexity of literature work. There are so many that it would take the entire thesis to describe and propose examples for each one.

Luckily that is not the objective of this thesis, so for the sake of simplicity, we will present just the main categories in which these features can be grouped: Lexical, Syntactic, and Semantic.

2.2.1 Lexical Features

The Lexical feature is the first that comes to mind when thinking about the complexity of literature work. The reason is the vocabulary of the work; everyone can imagine that a concept introduced using simple terms turns out to be easier than the same concept presented with an archaic or polished terminology.

Let us consider for example the two following passages about Mourning Dove:

> Mourning doves are light grey and brown, and males and females look similar. The species mostly have one partner at a time. Both parents incubate and care for their chicks. Adult mourning doves usually eat only seeds. The parents feed crop milk to the young. (simple Wikipedia)

> Mourning doves are light grey and brown and generally muted in color. Males and females are similar in appearance. The species is generally monogamous. Both parents incubate and care for the young. Mourning doves eat almost exclusively seeds, but the young are fed crop milk by their parents. (standard Wikipedia)

It is evident that the two passages manifest the same content but differ in terminology, with the passage from simple wiki containing less complex terms

compared to the regular version. Obvious is, for example, the use of “monogamous” in the standard variant versus “one partner at a time” in the simplified sentence. Or the use of “eat almost exclusively” compared to a simpler “usually eat only”.

2.2.2 Syntactic Features

Parallel to the lexicon used, the other element that jumps to mind, is the grammatical structure of the sentence, or more in general, the Syntactic Features of the document. While with lexical features we mainly focus on the nature of words, with syntactic features, we have to consider the role that every word plays in the sentence.

Broadly speaking, when considering Syntactic Features, we are referring to information that can be captured using either Part Of Speech tagging, Morphology, Chunking, Parse Tree, Dependency Analysis, or Analysis of Clauses.

Part Of Speech tagging is an activity that assigns to every lexical item a specific category, called, *part of speech* (POS). This tagging activity can be executed at two levels: Universal Part Of Speech (UPOS) and language-specific part of speech (XPOS) where the former is a coarse classification, language-independent, that assigns to every word one of the tags presented in table 2.1, while the second is a fine-grained classification, with a notation dependent by the treebank in use. Independently by the granularity of choice, the objective of this classification is to group all the words that play a similar role in the grammatical structure of the sentence. An example of POS tagging can be seen in Table 2.2 and 2.3, respectively for UPOS and XPOS.

Morphology, instead, is an association of a set of features representing lexical and grammatical property to a specific word form. Such features are additional information that goes in support to a POS tag by specifying either lexical information or the inflection of the considered word. Given their nature are usually divided into two big categories: Lexical Features and Inflectional Features. The former defines all the attributes associated with lexemes or lemmas, the latter, instead, describe all the information that depends on the form of the word. Every feature is usually presented in form Name: Value and every word can present multiple features; furthermore, features are language and treebank dependent. Table 2.4 displays an example of morphological tagging.

Chunking also referred to as “Shallow Parsing” is a technique to analyze sentences, that groups part of the sentence in higher-order units with a specific meaning, as it could be with noun phrases or verb phrases. This technique can either be applied by simply implementing classical research patterns, or using more advanced machine

Table 2.1: Table presenting the list in alphabetical order of the Universal POS tags

Tag	Meaning
ADJ	Adjective
ADP	Adposition
ADV	Adverb
AUX	Auxiliary
CCONJ	Coordinating Conjunction
DET	Determiner
INTJ	Interjection
NOUN	Noun
NUM	Numeral
PART	Particle
PRON	Pronoun
PRONP	Proper Noun
PUNCT	Punctuation
SCONJ	Subordinating Conjunction
SYM	Symbol
VERB	Verb
X	Other

Table 2.2: UPOS tagging of the phrase "Mary likes dogs and cats"

Mary	*likes*	*dogs*	*and*	*cats*
PROPN	VERB	NOUN	CCONJ	NOUN
proper noun	verb	noun	coordinating conjunction	noun

Table 2.3: XPOS tagging of the phrase "Mary likes dogs and cats"

Mary	*likes*	*dogs*	*and*	*cats*
NNP	VBZ	NNS	CC	NNS
proper noun singular	verb 3rd person singular present	noun plural	coordinating conjunction complementary	noun plural

Table 2.4: Morphological analysis of the phrase "Mary likes dogs and cats"

Mary		*likes*		*dogs*	
Noun Type:	Proper	Verb Form:	Finite	Noun Type:	Normal
Number:	Singular	Tense:	Present	Number:	Plural
		Person:	Third		
		Number:	Singular		

and		*cats*	
Conjunction Type:	Complementary	Noun Type:	Normal
		Number:	Plural

learning-based methodologies.

The strong point of this technique, in particular the variant based on machine learning, is that it can grasp also contextual information while composing the chunks, allowing for better classification, and solving the problem that combination of base elements might have different high-level meaning.

Parse Tree is an ordered rooted tree used to represent the syntactic structure of text input, according to some specific context-free grammar. in NLP, such grammar is a set of rules associated with a specific language, and leveraging such rules, the parse tree decomposes the structure of the sentence up to the single syntactic element.

Dependency Analysis is a powerful tool usually used to analyze documents written in languages in which the order of the words is not fixed based on their role in the sentence. Using the dependencies is possible to determine the connection among sections of the sentence without knowing the role of the words between them. Usually, the dependency is presented using a tree representation in which every node represents a word, and every branch or arrow is a link highlighting the relationship between the head (starting point) and the modifier (ending point). Dependency Models is the name the researchers use to describe this design, and Figure 2.1 exposes an example of such a model.

The strong point of this design is the simplicity with which is possible to recognize which words are governing over the others and which words instead act as simple modifiers. Relevant is also the lack of intersections among the branches; this is not a choice of design, but a property of the tree itself. This property holds if all the words are presented in a straight line and in the same order in which they appear in the sentence.

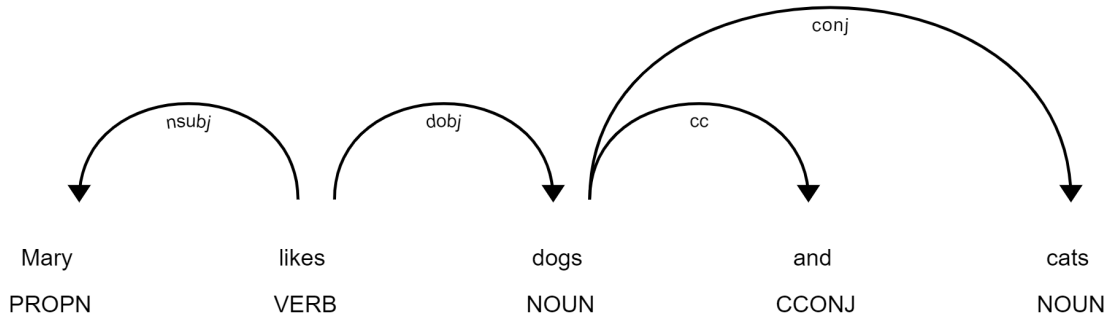


Figure 2.1: Dependency Tree structure of the phrase "Mary likes dogs and cats"

Analysis of Clauses, lastly, is used to refer to all the studies of the features of the clauses in a text, such as frequency of appearance or ratio. In this context, a clause is essentially a part of the sentence that contains a verb, usually, the "standard clause" is either made of a subject and a predicate or a verb with objects and modifiers. Generally speaking, a simple sentence indicates a single finite clause, while more complex sentences can include multiple clauses. It is, however, important to notice that sentence does not imply clause since the definition of the two concepts is not related.

When analyzing clause, two major distinctions are applied: main clauses versus subordinate clauses and finite clauses versus non-finite clauses. The main difference between the first two alternatives is the capability of the clause to constitute a complete sentence by itself or not; in this sense, a subordinate clause will always be dependent on the main clause. For what concerns the second pair, instead, the difference residing in the verb being either finite or non-finite.

Apart from these general classes, clauses can be sub-classified according to a specific trait that is dominant in the syntactic form, such as the position of the verb or the appearance of specific words (e.g. wh-word).

2.2.3 Semantic Features

After considering the nature of the words and the structure of the sentences, only one aspect remains when considering the complexity of a document: the Semantic of the document. Broadly speaking, it is possible to divide semantic features into two main categories: *semantic concepts* and *coherence*.

Semantic concepts cover the relationship between a component of the text, (either it be words, phrases, signs, or symbols) and the meaning or denotation associated. The main and most intuitive examples of this relationship are figures of speech, word plays, or even irony. In all these techniques is evident how through an alteration of the normal order of the terms in a sentence or the matching among words usually uncorrelated or opposite, the author can let the words carry a different

Table 2.5: Examples of Semantic concepts and relative explanation

Example	Type	Explanation
Economical with the truth.	Euphemism	Mild term used to substitute the more offensive "liar"
The Titanic was said to be unsinkable but sank on its first voyage.	Situational Irony	Contrast between the ideal and reality of facts
Heart of stone	Metaphor	Use of "stone" to describe a behavior and not the material
Parting is such sweet sorrow	Oxymoron	Parting can hurt people, yet it can also intensifies their feelings
Spilling that glue made a real sticky situation!	Pun	Uses glue's main property (stickiness) to make a joke out of the common phrase "sticky situation", which means a difficult situation.

meaning compared to the ordinary one. Some examples are presented in Table 2.5.

In this sense, it is immediately evident that the higher is the number and complexity of the "alteration of meaning" proposed in the text by the author, the higher will be the complexity of the work.

The second big family of Semantic Features is associated with the concept of coherence and cohesion and is analyzed using a technique called Discourse Analysis. Discourse Analysis (DA) is the field of linguistic that studies the coherence and cohesion of the text beyond the limit of the simple sentence, hence by considering the document in its entirety or examining only the portions associated with a specific character of the work. DA can be mainly implemented in two variants: locally or globally. The difference resides in the dimension of the scope of analysis, with the local study limited to linked or connected sentences, while the global analysis is applied to the entirety of the work. Out of all the features, only coherence and cohesion can verify the "meaningfulness" of the document analyzed, while all the other elements test the "correctness" of the sentence.

To better comprehend the relevance of coherence and cohesion in assessing the complexity of a document, let us consider these two passages about dogs:

> Dogs are canines that people domesticated a long time ago. Wolves are predecessors of dogs and they help people in a variety of ways. There are various reasons for owning a dog, and the most important is

> companionship. (Enago Academy)
>
> Dogs are canines that people domesticated a long time ago, primarily for practical reasons. Even though dogs descended from wolves, they are tame and can be kept in households. Since they are tame, people have various reasons for owning a dog, such as companionship. (Enago Academy)

The two passages offer the same amount of information; however, the second segment appears smoother and simpler to understand compared to the first. It is specifically this difference that causes an increase in complexity. A passage with a lower coherence or cohesion, force the reader to make up for the missing parts of the information, generating an increment in the mental burden. Meanwhile, a passage with high cohesion and coherence will display all the information logically and continuously.

2.3 Traditional Approaches

The first relevant study on Lexical Complexity appeared in 1948, by the hand of Flesch Rudolph in [9] who developed one of the most famous formulae for automatic readability assessment of documents. The formula was quite simple and based on superficial features of the text, such as the total number of syllables, the total number of words, and the total number of sentences. The formula was computed as follow:

$$score = 206.835 - 1.015\left(\frac{\text{total words}}{\text{total sentences}}\right) - 84.6\left(\frac{\text{total syllables}}{\text{total words}}\right) \tag{2.1}$$

Where the resulting score is a position on a scale that goes from 0 (difficult) to 100 (easy), it is due notice however that the extreme values retrieved using this scale are -515.1 and 121.22 were the former is assigned to a sentence at the beginning of [10] and the latter achieved only if every sentence consists of only one one-syllable word.

Equation 2.1 got so much notoriety up to the point of being used still today in many systems such as Microsoft Office Word[1] or Grammarly[2].

This initial formula proposed in [9] has been further revised in [11] where an alternative version, tailored for the US school system was proposed, making it easier for teachers to properly assess the complexity levels of the literary works. The new

[1] https://www.microsoft.com/en-us/microsoft-365/word
[2] https://app.grammarly.com/

variant was computed as follow:

$$score = 0.39 \left(\frac{\text{total words}}{\text{total sentences}} \right) + 11.8 \left(\frac{\text{total syllables}}{\text{total words}} \right) - 15.59 \tag{2.2}$$

Where score, in this case, is a number corresponding to a U.S grade level. It should be noted that given the different nature of the used weights, the two versions of the formula are not compatible.

Parallel to Equation 2.1, another famous approach was the one presented in [12] where the author proposed a formula for the computation of the grade level of a certain document. The formula was presented as follow:

$$score = 0.1579\text{PDW} + 0.0496\text{ASL} + 3.6365 \tag{2.3}$$

Where score has to be intended as the reading-grade score of a pupil at 3rd grade or below, PDW as Percentage of Difficult Words (Words outside the list of 3000 common English words, defined by Dale) and ASL as the average sentence length.

These formulas represented the standard for various years. More recent studies, however, highlighted the inaccuracy of such metrics for computing reading complexity [7, 8]. For instance, a problem is represented by the *concept* associated with the analyzed words, for example, as mentioned in [13], the word "quark" is simple to read but the associated meaning is very complex.

2.3.1 Approaches Based on Lexical Features

The first attempts to compute complexity based on lexical features were introduced by [14] that presented a statistical language model vocabulary-based approach. A statistical language model use information concerning pattern word usage, frequency, and order, returning a probability distribution of predictions.

Citing [15] "A statistical language model can be thought of as a word histogram giving the relative probability of seeing any given vocabulary word in a text." In particular, [14] presented a model able to capture specific information concerning word usage in different complexity levels. This allows the algorithm to process more data on the relative difficulty of words, compared to the previous approaches.

An improved approach appeared in the work of [16] and [17], in which the researchers proposed alternative statistical models. These models were also the first ones to be associated with the analysis of syntactical features of a text.

A similar approach, proposed by [18] and [19], is Word Maturity. Word Maturity Usage analyzes the handling of single words and phrases during the learning phase. Specifically, the system can consider both word usage and its context-dependent meaning.

All these approaches were directly or indirectly employing the frequency of the words as the core feature. However, before [20], no deeper analysis has been done on the frequency itself. They introduced a series of experiments aimed at determining the relevance of Lexical Complexity in the computation of readability. These experiments were based on frequency lists generated from subtitles for TV channels and cross-validated on the WeeBit and Appendix B of Common Core Standard corpus (AppBCCS). The results proved that normalized forms of the frequency, such as the Zipf measure, can provide a more stable outcome. Furthermore, the analysis demonstrated that the source of the sentences used for computing the frequency list deeply affects the results. For example, it was noticed how a frequency list derived from a channel for children was better suited to reflect the different cognitive burden from vocabulary retrieval typical of a school environment.

Effectiveness of the frequency as an estimator of text complexity has also been proved in [21], supported by the idea that the more frequent is the word, the lesser is the burden in retrieving it from the memory of the individual. An example of this behavior can be observed in [22], where, by substituting a third of the words with similar uncommon definitions, the Readability Score drastically decreases.

Parallel to these approaches based on frequency, in [23], a study centered on low-level lexical metrics was proposed. In the research, a total of 25 metrics were considered covering multiple aspects of Lexical Complexity, including Lexical Density, Lexical Sophistication, and Lexical Variation. These metrics were introduced in the literature between the '50s and '90s and acted as comparison data in the following researches.

These approaches are quite powerful because they allow us to obtain results with a high level of correlation with the Reading Complexity of a text; without requiring nor excessive analysis of the grammatical aspect of the language, neither high computational power.

Unfortunately, it is not easy to identify a feature able to describe the Lexical Complexity in every aspect, and even if one is defined, it is hard to represent the real vocabulary knowledge of the individual. For example, the features of the vocabulary of a person working in the medical sector will differ from an individual specialized in the economy field; in fact, every person presents a different knowledge about a language according to his living experience and environment.

Furthermore, the lexical features of a language tend to change with time, and in the current and multi-connected society, this happens even often, limiting the lifetime of the corpora used as a reference.

2.3.2 Approaches Based on Syntactic Features

The first analysis of Syntactic features as a means to compute Lexical Complexity was born following researches that highlighted the longer processing time associated with a longer and more complex structure of the text [24]. These researches initially focused on the structure of the sentence by highlighting the causal organization of the document. For this reason, works based on the usage of parse trees to categorize the clauses and analyze the ratio of appearance or the subordinate relationship made their appearance [25]. This evolution appeared parallel to the implementation of statistical models for the analysis of lexical features, and in [17] is possible to see the first attempt to match this two kind of features by including a POS tagging representation of words.

Further studies revealed, however, how it was not clear to which extent the syntactic information produced was helping in the computation of the complexity, the need to improve the quality of the information provided was so significant that a new technique called shallow parsing was introduced. This innovation allowed the researchers to comprehend the differences between the various clauses and to perform deeper investigations. An example of this evolution can be seen in [16], where the researchers evolved the approach proposed in [17] by completely distinguishing between the analysis of lexical and grammatical features. In particular, the researchers proposed a method, for the syntactic part, based on the study of the verbs, according to the results of a shallow parsing technique using a KNN algorithm.

Worth of notice is also the work presented in [26], where an important study involving different genres and features, namely lexical, syntactic, and language model-based, was able to obtain the highest level of correlation by merging all the traits.

Lastly, a quite interesting approach is also proposed in [27], in which 14 low-level metrics to measure Syntactic Complexity were presented. Such metrics can be grouped in five different categories: Length of the production unit, Sentence Complexity, Subordination, Coordination, and Particular structure. These metrics are a subset of large-scale research offered in [28] and further explored in [29].

Independently by the huge amount of research done on the topic, unfortunately, up to today, it is still considerably hard to identify which aspect of the syntactic peculiarities can better expose the difference between the levels of complexity. Often the results reported by studies contradict the ones presented before. The main reason for this discrepancy is probably associated with the intrinsic syntactic difference that distinguishes different genres of text, and the influence that the background of the individual can have on his writing style. Regarding this aspect,

various studies highlight the huge difference between the writing habit of native speaker and second language learners, with the second manifesting a structure with a heavy correlation to their native language. [30, 31]

Lastly, it is of interest that not always Syntactic Complexity can be treated as a perfect indicator of Text Quality. Various studies concerning learning material for specific languages, and passages composed by students, highlighted that not always a higher complexity implies a better quality of the text, as exposed in [30].

2.3.3 Approaches Based on Semantic Features

When considering approaches based on Semantic Features, we need to distinguish between the will of the researcher to focus either on semantic concepts or discourse analysis.

When talking about semantic concepts, there is only one framework that is relevant and immediately comes to mind: Coh-Metrix. Coh-Metrix is a linguistic tool presented in [32, 33] that had a relevant role in automatic readability assessment. This system, in particular, was able to provide a multi-dimensional set of 108 features (called indices) for text representation.

This system is both able to cover the analysis of the semantic concepts, by considering the concreteness, imageability and ambiguity of words against a database, and discourse analysis by capturing high-level features as:

- degree of referential cohesion (e.g. noun overlap of adjacent senteces)
- deep cohesion (links between cause-effect sentences)
- degree of narrativity (cohesion associated with story telling)
- temporality (degree of cohesion among tenses and temporal features)

If for semantic concepts the main point of reference is Coh-Metrix, for discourse analysis is possible to choose among multiple approaches.

The will to focus solely on discourse analysis, and in particular, coherence, started to make its appearance in early 2000, when experiments with real individuals tried to identify if it was possible to distinguish lexical cohesive patterns among the texts [34]. This experiment proved to be fruitful giving the start to a great interest in the topic.

One of the most relevant approaches was presented in [35], which introduce the Entity-Grid (EG), a statistical model based on Centering Theory [36]. The EG paradigm is based on the simple idea of continuously testing entities in sentences assigning them to a syntactic role. In this way, the algorithm can estimate the probability that an entity is the subject of the sentence given the previous elements

and roles. Needless to say, the software is simply evaluating the presence and extent of local coherence.

Even though this model was quite innovative, it was also basic and naive in some aspects, for this reason, researchers re-implemented the technique by adding further features to the model. An example of this idea can be seen in [37] where *discourse prominence*, *named entity categorization* and *individuation of relevant entities* were implemented in the pre-existing EG model.

More recent approaches, as natural as it would be, are instead more focused on neural systems that either decided to maintain the usage of EG models, such as [38] that fed the EG representation to a Convolutional Neural Network or strayed away from it, like in [39]. This last approach is based on the implementation of a local coherence model by encoding patters of changes in the semantic relationship between sentences. The main relevance of this approach is double founded, firstly, the ability to overcome the need of using external dependencies, such as the coreference resolution system, typical of an ED-based model, and secondly, the capacity to encodes the words based on their sentence context.

These new approaches proved to be effective in their task, even if perplexity is still present on the active correlation of Semantic Features and Complexity of a text, in particular, compared to the high level of correlation produced by Lexical and Syntactic Features towards Complexity.

2.3.4 Approaches Based on Multiple Features

In more recent works, researchers are moving toward a merging strategy, by creating systems based on all the previously mentioned type of features.

The first work in which this approach was implemented is [40], in which the scientist implemented an analysis based on readability factors such as vocabulary, syntax, cohesion, entity coherence, and discourse. They executed a study using a specialized news corpus created with texts from the Wall Street Journal intended for an educated adult audience. The various experiments concluded that the introduction of all traits greatly increased the accuracy and performance of the final model.

Relevant is also the work of [41], in which the researcher presented a deep analysis and comparison of various features to compute Lexical Complexity. In particular, after comparing the behavior of discourse, language modeling, parsed syntactic, POS-based, and shallow features, the authors found out that selecting the best trait for every group obtained the best result, scoring slightly better than the simple naive usage of all features.

The last work that we want to cover in this section is the enormous study

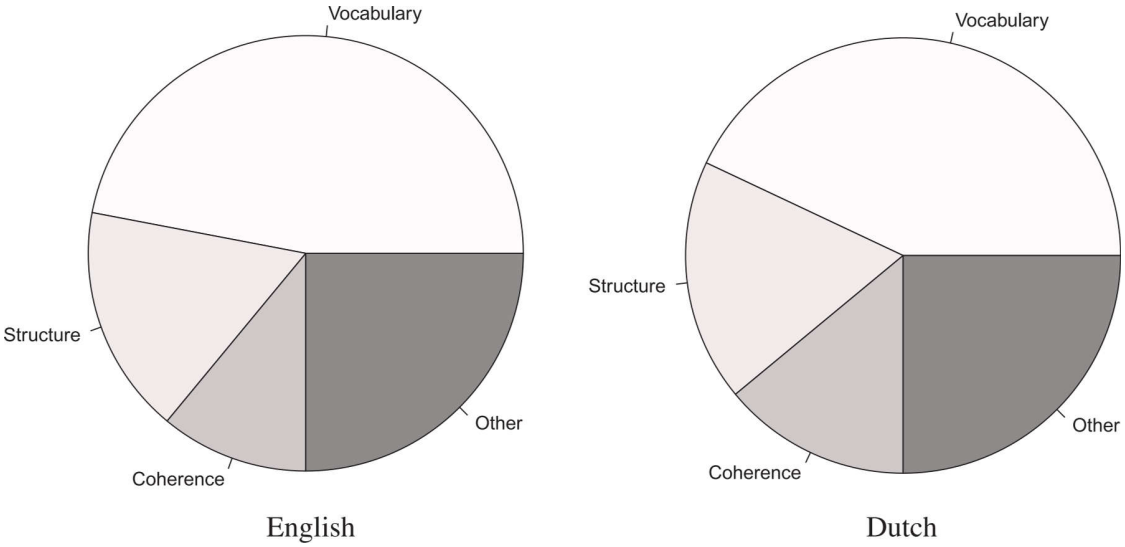


Figure 2.2: Pie charts representing the importance added to the various feature groups by the expert assessors based on their comments for both the English (left) and Dutch (right) dataset. Image was taken by [1]

executed in [1], in which the researchers tested the relevance of 87 features in computing text readability for English and Dutch. The particularity of this work resides in the way these features were selected, a pool of experts reviewed every document and highlighted the elements that generated the difference in complexity. The authors also describe the annotation taken by the experts and divided them into four categories: Vocabulary, Structure, Coherence, Other, where "Other" represent comments like "I had to read the passage multiple times". The classification, presented in Figure 2.2, shows the relevance of comments concerning vocabulary compared to structure and coherence.

In this research, the results obtained were impressive; however, the researchers based the work on engineered features tuned for the two languages of choice. Given the wide variety of languages available, this approach is limited in terms of scalability and adaptability. Nevertheless, the experiment is useful because the outcomes denoted the features that characterize a specific language.

2.4 Background Knowledge for Deep Learning Approaches

In this section, we are going to present some background knowledge needed to understand the most relevant and recent approaches in computing reading complexity using deep learning. When considering these innovative approaches, there are three main concepts that one must know of: Embedding, RNN (in

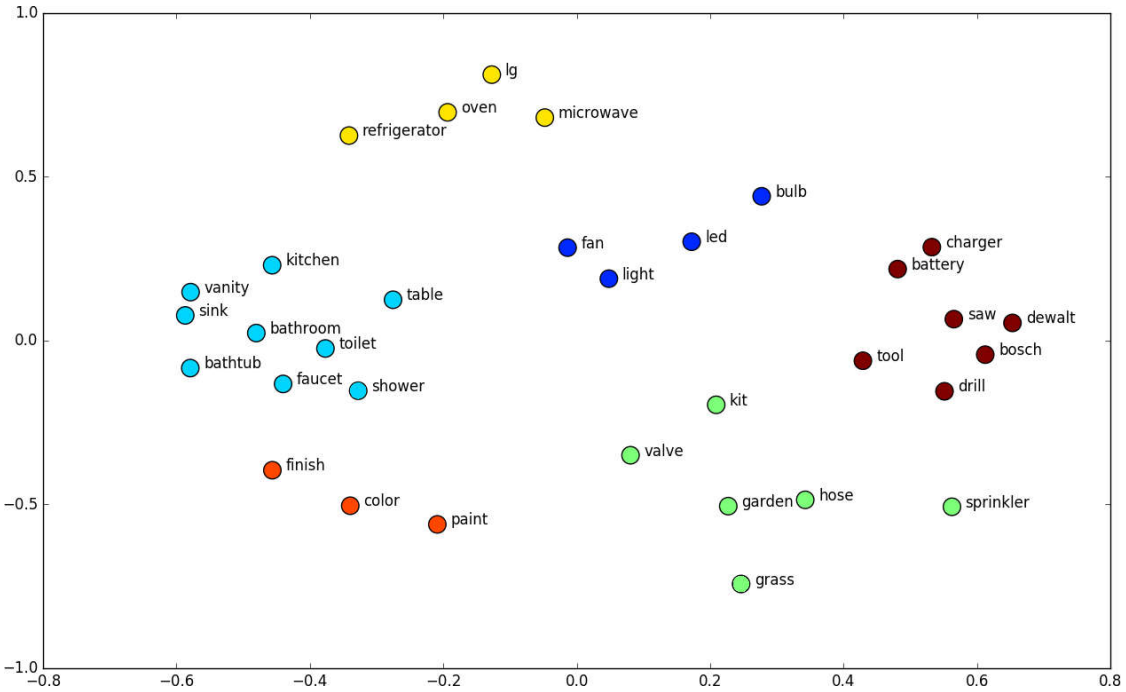


Figure 2.3: Example of 2D Embedding of some words

particular LSTM), and Attention.

2.4.1 Embedding

Generally speaking, an embedding is a relatively low-dimensional space into which translates high-dimensional vectors. Embeddings were born to ease the handle of large inputs of data, in particular sparse vectors, when applying machine learning procedures. Ideally, an embedding mechanism can maintain only relevant traits of the input. This action will lower the dimensionality, meanwhile granting that semantically similar inputs will be close in the embedding space.

Embedding is a technique with application in multiple fields; the one used in NLP is called Word Embedding. Word embedding is a procedure in which words or phrases are mapped to vectors of real numbers. Conceptually speaking, it is a mathematical embedding from space with many dimensions per word to a continuous vector space with a significantly lower dimension.

Embeddings can be generated using a multitude of methods, the main used ones are neural network feature extraction, dimensionality reduction based on word co-occurrence matrix, and probabilistic models.

This technique demonstrated to be very efficient, if applied to the input of a neural network model, by greatly boosting the performance in Natural Language Processing tasks.

Figure 2.3 displays an example of 2-Dimentiosnal embedding representation,

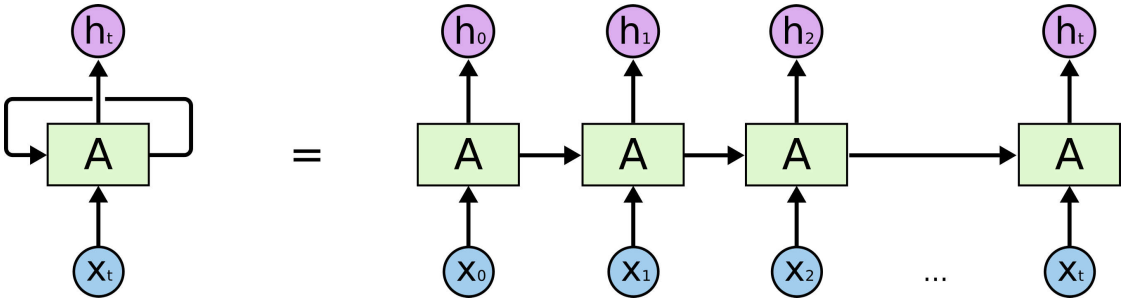


Figure 2.4: Example of unrolled RNN

where it is evident the ability of the mechanism to maintain a relationship among the input data.

2.4.2 Recurrent Neural Network

Recurrent Neural Networks (or RNN) are an approach theorized during the '80s to solve the inability of neural networks to parse a sequence of data retaining information trough time. This approach was inspired by the human mind; imagine, for example, to buy a new book and to open it at a random page in the middle of the work and to start reading. It is highly probable that you will not be able to understand who are the characters, which is the location, which is the course of events, and multiple other information. This is happening because you have no knowledge about the events before the current moment of the story. If instead, you reach that instant of the story, starting from the beginning of the book, it is highly probable that you will be able to grasp all the information previously unavailable.

Following exactly this line of thought, RNNs were created to retain information by introducing a looping system that allows the data to persist in the network. Figure 2.4 displays a representation of this model with the core structure both in the real and the unrolled version. Looking at the unrolled version is possible to notice the strong relationship between this network and sequence like data.

As we highlighted, the main feature of RNNs is to maintain information through time, allowing that knowledge to influence the current decision, but is it always true? Well, the answer is it depends.

If the information that we are trying to retrieve is recent then RNNs are perfectly able to use it, unfortunately, the same is not true if the information is way back in the text, in fact, the larger is the gap between the instant in which the useful information was proposed and the moment in which is required, the more difficult is for the network to maintain it. This problem is referred to as "Long-Term Dependencies Problem" and was deeply covered in [42], in which the researcher proposed the assonance between the mentioned problem and the more famous

"Vanishing Gradient Problem" that appears in neural networks.

To solve this problem, Long Short Term Memory networks (or LSTMs) were proposed in [43] and further evolved by other authors. This new kind of network made the ability to retain long term information their forte, by changing the inner structure of the repeating module, passing from one simple layer to four interacting layers.

Figure 2.5 portraits this evolution. In the image, every line carries an entire vector that can either be concatenated with other vectors or being duplicated. The pink circles indicate a point-wise operation (multiplication, addition, and hyperbolic tangent), while the yellow boxed a neural activation layer (sigma activation function and hyperbolic tangent).

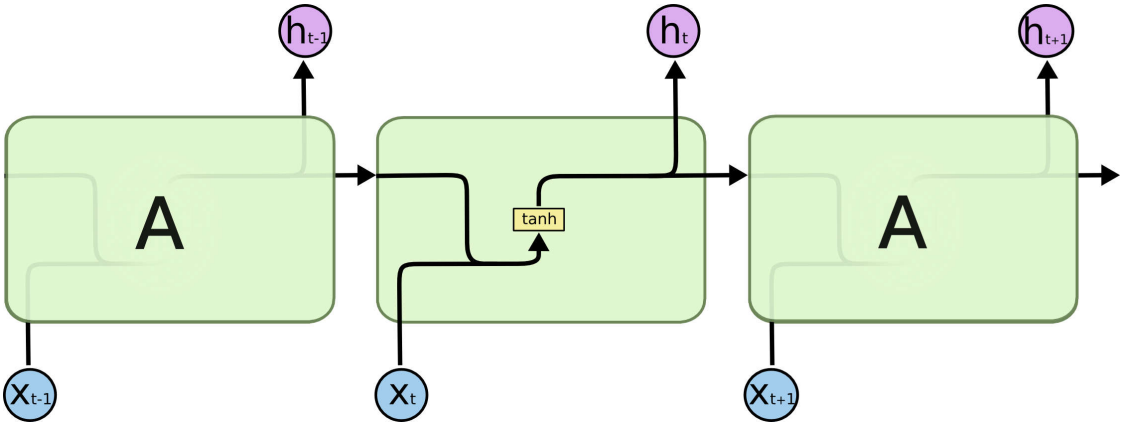
Considering the image, we can notice that firstly, the cell state (represented by the upper horizontal line) runs through the entire chain system with minimum interaction, granting that information are retained unvaried. Secondly, the network can alter the cell state using gates to regulate the necessary changes. Figure 2.6 displays a representation of an LSTM cell with the associated formula.

Some variants of LSTMs have been proposed in the recent years, out of all the introduced ones, the most relevant is Gated Recurrent Unit (GRU) presented in [44], that merges the *forget gate* and *input gates* into an *update gate*, while *cell state* and *hidden state* into a unique state. The resulting model is simpler and faster to train compared to LSTM but theoretically can maintain information for less time. Figure 2.7 illustrates a cell of a GRU network.

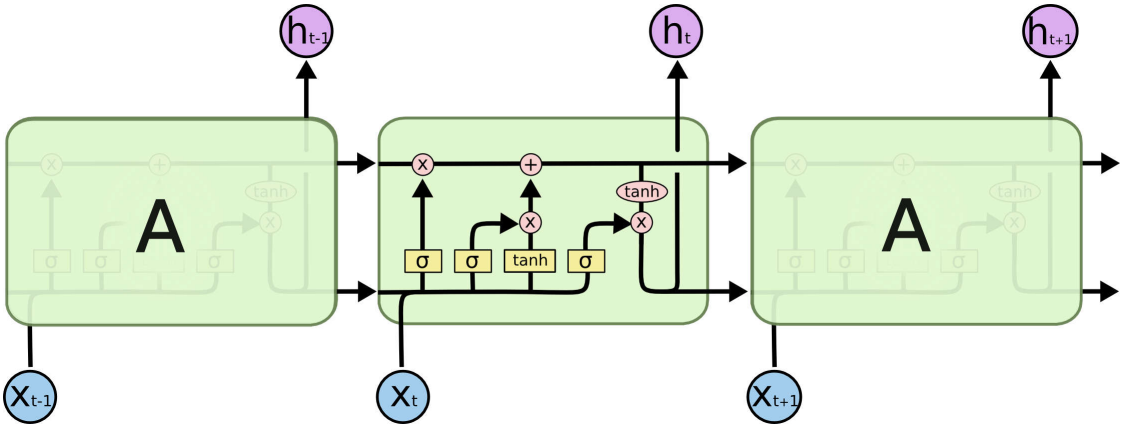
2.4.3 Attention Mechanism

The attention mechanism is an attempt to implement the equivalent of human focusing ability on a few relevant aspects while ignoring others when analyzing content. Imagine, for example, to look at your school class photos and someone asks: "how many people are there?". Instinctively you are going to look for some peculiarities that allow you to count the number of people in the image; most of the people will count by looking at how many human heads they can see. This decision is the result of a selection made by the brain to remove redundant or useless information for the task at hand, so to focus only on the minimum information needed. In the same way, the objective of attention in a neural network is to select what is more relevant for the completion of the task.

The attention mechanism appeared for the first time in [2], where the researchers proposed an innovative encoder-decoder based neural machine translation system. An encoder-decoder translation system can be basically considered as two RNNs, the first, called an encoder, reads the input and tries to make sense out of it, then



(a) Structure of a simple RNN



(b) Structure of a classic LSTM

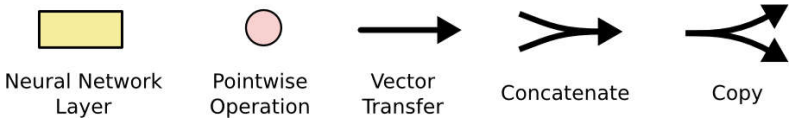
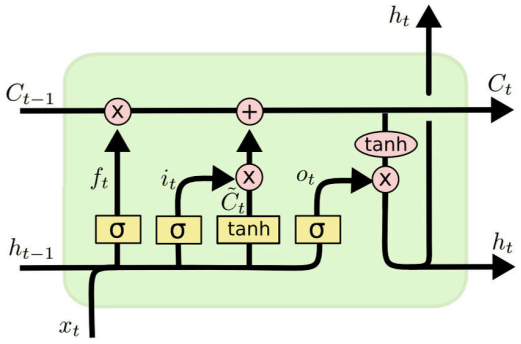
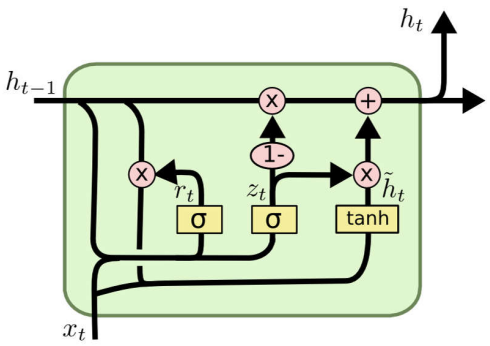


Figure 2.5: Comparison between a simple RNN cell structure and an LSTM one



$$
\begin{aligned}
f_t &= \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right) \\
i_t &= \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right) \\
\tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C) \\
C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\
o_t &= \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right) \\
h_t &= o_t * \tanh\left(C_t\right)
\end{aligned}
$$

Figure 2.6: Cell structure of an LSTM network



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure 2.7: Cell structure of a GRU network

pass the generated summary to the second network, called decoder, that applies the information for the translation of a new sentence.

Given this structure, it is evident that the quality of the translation will be dependent on the quality of the information retrieved by the first network, and due to the Long-Term Dependencies Problem (see Section 2.4.1), the result with long sentences is inadequate. A possible solution to this problem is to reduce the amount of information maintained at every step and a system able to identify the most relevant data, perfectly satisfies this requirement.

Figure 2.8 displays the first attention mechanism, in which, to every hidden layer of the Bidirectional LSTM (See 2.5.1) is applied a multiplication by a factor α_{ij} and then summed to retrieved the final context vector c_i or in formula:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{2.4}$$

where α_{ij} are the weights computed as softmax of the output score of a feedforward neural network that attempts to compare the alignment between input at j and output at i:

$$\alpha_{ij} = \frac{\exp\left(score_{ij}\right)}{\sum_{k=1}^{T_x} \exp\left(score_{ik}\right)} \tag{2.5}$$
$$score_{ij} = \text{ff-net}(s_{i-1}, h_j) \tag{2.6}$$

The researchers defined this kind of attention as "Additive" and it represents only one of the various versions that appeared on the research panorama in recent years. Attention can be classified according to their nature, unfortunately, given the huge amount of different variants proposed and the intrinsic differences depending on the type of task, it will be of poor interest for the current thesis to go in excessive details.

For this reason, we are simply going to propose the general and broad categories in which attention mechanisms can be organized.

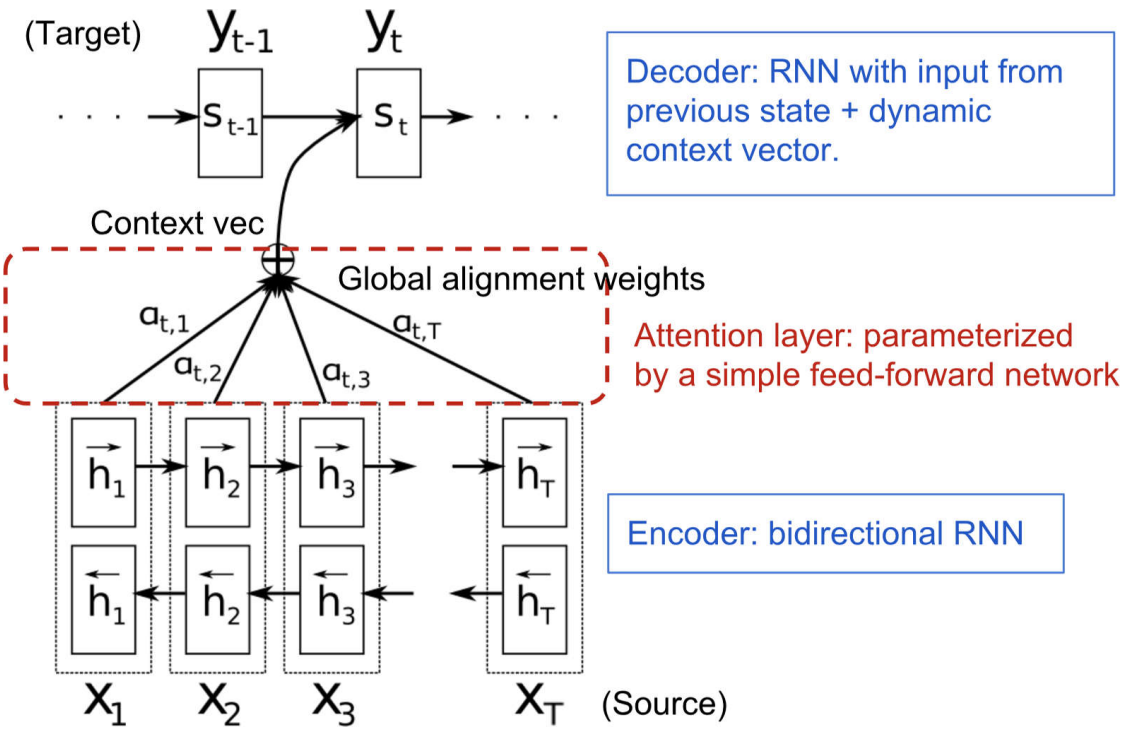


Figure 2.8: The encoder-decoder model with additive attention mechanism in [2].

Self-Attention Self-attention or intra-attention, is a mechanism that compares different sections of a single sequence input. It proved to be effective in tasks such as machine reading, abstractive summarization, or image description generation. Figure 2.9, extracted from [3], provides an example of an application. In this research, the attention mechanism was used to learn the correlation between the current words and the previous part of the sentence.

Global/Soft Attention Global or Soft Attention is a system in which the alignment weights are learned and placed over the entire input sequence. The notation was first proposed in [45, 46] and also the mechanism proposed in [2], fall in this category.

Hard Attention Hard Attention was proposed in [45] as the opposite of Soft Attention, and the main difference relays in the scope of application, with the former being focused only on part of the input sequence.

Local Attention Local Attention was proposed in [46] as the opposite of Global Attention, and can be taught of as a blending between Soft and Hard Attention, with the improvements being: the initial prediction of a single aligned position for the current target word, and then use a window centered in the source position to compute the context vector.

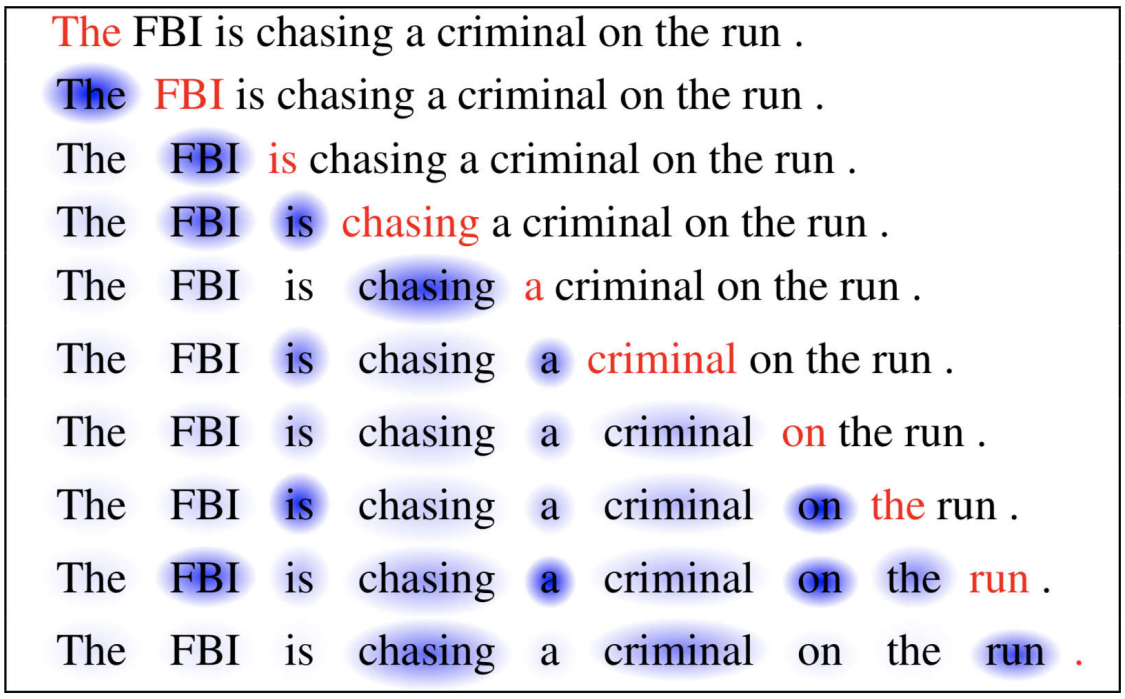


Figure 2.9: Example of Self-Attention taken from [3], in which the current word is in red and the size of the blue shade indicates the activation level.

2.5 Approaches Based on Deep Learning

In the most recent years, the advent of machine learning and deep learning was also felt in NLP tasks, with a great increase of methodologies based on these innovative techniques.

In the course of this section, we are going to analyze mainly two papers, both published this year. The first summarizes the state of the art for supervised learning approaches for text classification, the second, proposes an interesting approach, that will be used as inspiration and base for part of the thesis work.

2.5.1 Paper #1: Supervised and unsupervised neural approaches to text readability

The first paper, being [47], executes an analysis implementing three different algorithms, that are considered as the most innovative in recent years, namely: Bidirectional Long Short Term Memory Networks (BiLSTM), Hierarchical Attention Neural Network (HAN) and Transfer Learning using BERT.

Models

BiLSTM is an approach introduced in [48] where a new variant of LSTM is proposed, by implementing a concatenation of forward and backward LSTM layers to read the documents in two different directions, the resulting feature matrix of the LSTM is then processed by applying max and mean pooling. The obtained maximum and average values vectors are then concatenated and fed to a linear layer that generates the final result.

HAN is a technique introduced in [6] based on the idea that words make sentences and sentences make documents, however not all words are equally important in a sentence, for this reason, to some of them must be provided a specific weight that is higher than others. For both word and sentence level, the executed procedure is the same, the input is processed using stacked recurrent neural networks, followed by attention model to highlight the most relevant data that is lastly, congregated to generate a vector. The generated vector will either be the input for the sentence level, if it is coming from the word level, or it will be used in defining the final classification if generated at the sentence level. The double level attention mechanism is a direct implementation of what proposed in [2, 45].

Transfer Learning, lastly, is a technique based on the use of a model developed for a different task, as the starting point for the task at hand. This approach has gained popularity in NLP and Computer Vision tasks thanks to the initial advantage that they provide; in particular, taking into consideration the enormous amount of computational power needed to train these models. One such example is Bidirectional Encoder Representations from Transformers (BERT), developed by Google in 2018 [4].

BERT's key innovation consists of the application of bidirectional training of transformers to language modeling, in particular, the researchers implemented a new technique called Masked LM (MLM), that is applied before feeding the word sequence to the model. When applying MLM, 15% of the words in each sequence, are replaced by the token "MASK", the model will then try to predict the original value of the tokens; BERT's loss function will then consider only the prediction generated by said tokens, ignoring the others. Furthermore, during training, the BERT model, will receive pairs of sentences and will try to learn if the second sentence is the real subsequent sentence of the first one, executing what the researches defined: Next Sentence Prediction (NSP).

Figure 2.10 displays the overall pre-training and fine-tuning procedures for BERT, as described in [4].

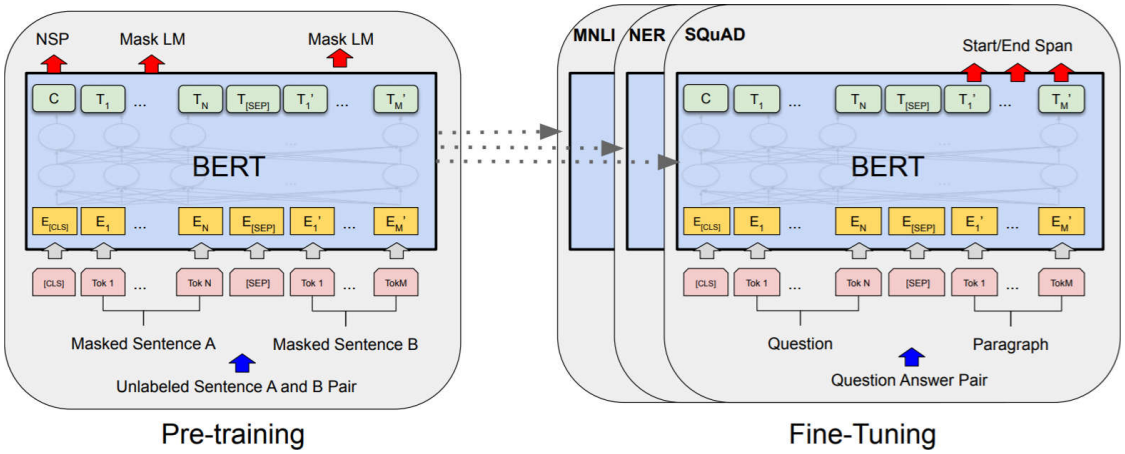


Figure 2.10: Overall pre-training and fine-tuning procedures for BERT [4]

Corpora

The structure used in the paper is composed of 12 layers of size 768 with 12 self-attention heads.

All three models were tested using three English corpora for text classification purposes:

WeeBit a corpus made of articles form WeeklyReader and BBC-Bitesize, classified in 5 groups according to the age of the target audience for a total of 3000 documents, 600 per class.

OneStopEnglish a corpus organized in 3 levels containing articles written specifically for English as Second Language (ESL) learners for a total of 189 texts provided in three versions, one per level.

Newsela a corpus corresponding to the age and classification proposed in the American school system from grade 2 to 12 for a total of 1911 original English articles with up to 6 versions for a total of 9565 English documents organized unequally in 11 classes.

Approach

These datasets were used after applying a classical subdivision, 80% training set, 10% validation, and 10% testing for Newsela and a five-fold approach with the same ratio for OneStopEnglish and WeeBit. The last two datasets are considerably smaller then Newsela, and similar data division would be invalidating.

BiLSTM and HAN were trained for a maximum of 100 epoch, then the best performing model on the validation set was used to predict the result on the test set. BERT, instead, was tuned to the job at hand for 3 epochs, after being trained

on a text understanding task.

From the experimental results, the researchers noticed that the performances of the various classifier were strongly varying according to the dataset used. In particular, BERT demonstrates the highest amount of variations, probably influenced by the typology of the task on which it was pre-trained, making it more sensible to semantic information.

HAN and BiLSTM demonstrated instead a more stable outcome with the former outperforming the latter in OneStopEnglish and Newsela while lagging a little behind on WeeBit. Given the nature of the corpora in which HAN performs better, it can be hypothesized that it can better identify syntactic and structural information then semantic ones.

2.5.2 Paper #2: Multiattentive Recurrent Neural Network Architecture for Multilingual Readability Assessment

The other work we wanted to present is [5], in which an innovative and interesting way to process text complexity is applied: multi-attentive recurrent neural network. While the internal core structure can be defined as classical, since it is attention-based BiLSTM, the innovative part reside in the decision to process each variety of input in a separate way.

Contrary to the standard approach, instead of providing the network with simple text, the input is enriched by associating also UPOS and morphological information. These three inputs are fed to the network parallel and independently. After passing the first network, to every type of data is applied an attention mechanism that consists of two layers neural network; describes as follow:

$$att_{a1,ij} = \sigma(w_1 \cdot \omega_{a,ij} + b_1) \tag{2.7}$$

$$att_{a2,ij} = \sigma(w_2 \cdot att_{a1,ij} + b_2) \tag{2.8}$$

where w and b are respectively the weights and bias of the layers, $\omega_{a,ij}$ represent the input to the network, with $a \in \{\text{text, upos, morph}\} = A_t$ indicating the attention type and ij referring to word j of sentence i, and lastly, σ represent a sigmoid activation function. Every attention is then multiplied by weighted value $z_{a,norm}$ automatically estimated during the training phase, such that $\sum_{a \in A_t} z_{a,norm} = 1$. This condition is forced by applying a softmax to the value of z:

$$z_{a,norm} = \frac{\exp(z_a)}{\sum_{a \in A_t} \exp(z_a)} \tag{2.9}$$

Figure 2.11 displays a graphic representation of the process.

The decision to implement this methodology, by using a multi-attentive system, is based on the belief that not all syntactic features are equally important in

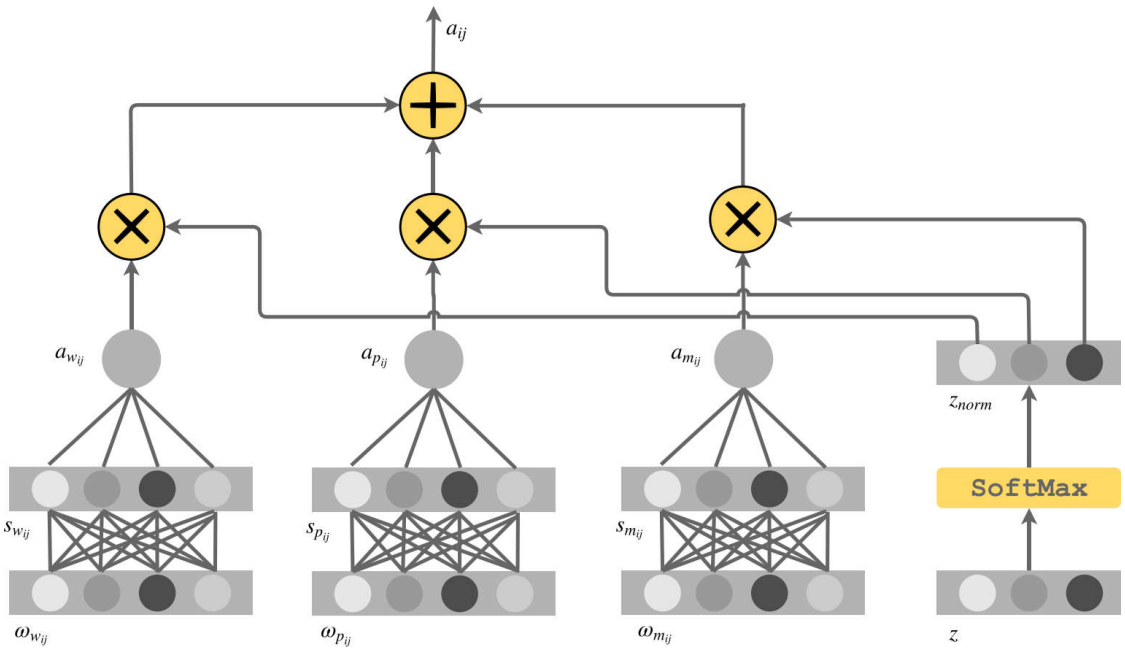


Figure 2.11: Example of the multi-attentive attention mechanism for word j in sentence i, image form [5]

computing the complexity, hence also the attention mechanism has to consider their relevance.

The same approach is then executed for the attention at the sentence level, using as input (ω) the last hidden state of independent BiLSTM, and the final prediction will be the result of the output of the RNN on the embedded text, multiplied by the two computed attentions.

This network was tested with a total of 5 corpora spanning multiple languages, however, for most of the corpus, the system proved to be inefficient and inferior compared to the ones presented in the first paper [47]. Nevertheless, we considered the approach interesting because it allows the creation of a network specialized in analyzing only specific traits of the document.

Chapter 3

Rationale behind the Work and Datasets

3.1 Rationale

Chapter 2 presented the most relevant and recent approaches developed for computing Reading Complexity, however, independently by the features or techniques used, the final objective was always to compute a single value able to assess the general complexity of a text. We can further improve the approach by presenting distinct classification systems that will take into account the real nature of the analyzed features. While, in general, it can be considered a good approximation to define a general Reading Complexity, in some cases, the contribution provided by the different sub-types of complexity can be very different and variegate. (See Section 1.2)

It is possible to observe a clear example of this behavior in passages that accentuate one aspect of complexity while oversimplifying another; For example, a simple sentence with difficult words in it, such as:

He will abjure his allegiance to the king.

will be normally classified either at medium or high complexity, while this is correct for the difficulty of the lexicon used, it is misleading from a pure syntactic point of view; syntactically speaking, the sentence is simple with a low complexity level.

Our research tries to assess exactly this problem by introducing BASILISCo, an approach that can compute both Lexical and Syntactic Complexity, independently by the level of the reader. The two complexities will be uncorrelated but will possess a strong implementation coherence granted that they will be estimated similarly, following the same rationale and workflow.

The workflow is presented in Figure 3.1 and can be summed up into 6 phases:

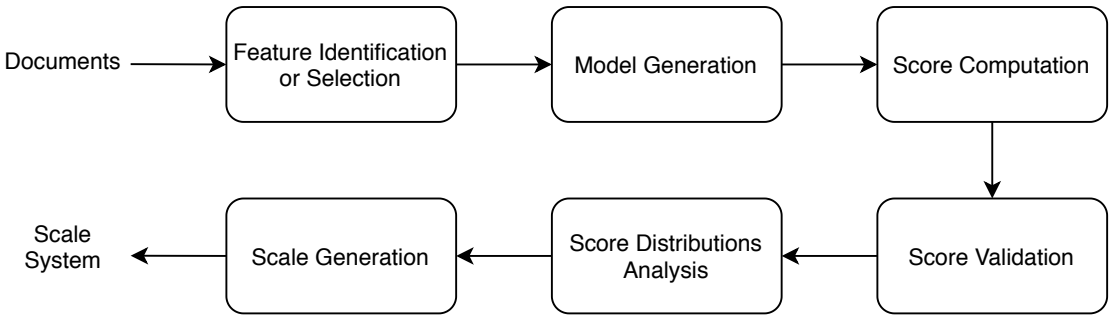


Figure 3.1: General workflow of the process

Features Identification or Selection: This phase is very important and, as the name suggests, consists of the identification and selection of the features that the next step will manipulate.

Model Generation: As the name implies, this section is responsible for defining a model that will be used as a reference system to compute the score.

Score Computation: In this part the complexity score is computed, using the model generated in the previous phase.

Score Validation: In this phase, the score is validated trough correlation analysis with low levels metrics presented in previous approaches, to determine either lexical or Syntactic Complexity.

Score Distributions Analysis: Once computed the complexity scores, for every text sample, the results are divided by complexity level and examined to identify the continuous function that better represents the distribution of the data.

Scale Generation: Lastly, this phase is responsible to provide a semantic to the generated score, by defining a scale divided in groups.

These steps represent the general workflow that will be followed, in a slightly adapted version, during the computation for both Lexical and Syntactic Complexity.

Since we aim to present a decomposition of the Reading Complexity into some of the underlying categories, a need to grant a coherent classification system is required. To achieve this, we believe that the best strategy is to maintain consistency throughout the entire generation process, from the first to the last step.

An obvious side and adverse effect of devising a process in this way is the standardization of the approach itself; this constraint might limit the possibility of creating a specialized scaling system. However, we concluded that it is necessary to grant a strong cohesion in the created scale; in this sense, the positive aspects greatly overcome the negative ones.

3.2 Datasets

Considering the nature and aims of our thesis, we wanted to test our system using a series of datasets that reflect a general knowledge context with a standardized metric system, possibly spanning multiple levels.

Considering these requirements, we opted for datasets organized according to the Common Core Standard system, introduced in the USA to regulate and identify the knowledge level that each individual must possess at different school grades.

The US school system is organized in a total of 14 grades, in which each grade roughly translates to one year of life, between 3 and 17 years. This period covers the instruction time that goes from nursery school to the last year of high school. The first two grades are PK and K, (Previous Kindergarten and Kindergarten), while the following ones span from 1 to 12. Seldom an additional category is established (CCR - College or Career Ready) and is employed to describe all the individuals with a level higher than 12.

A visual representation of this organization can be seen in Figure 3.2.

This system covers various aspects of the knowledge that a student can achieve in his school life; among them, one of the key metrics represents the capability of comprehending written text.

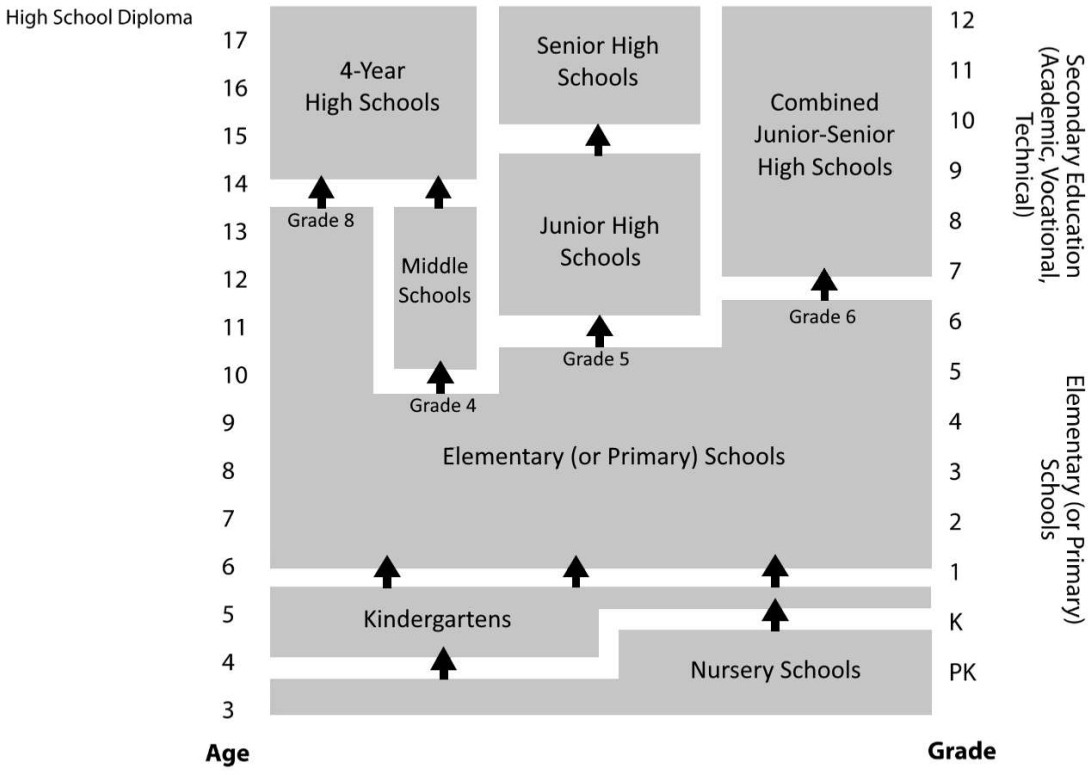


Figure 3.2: Schema of the US school system

We implemented three datasets that propose a classification organized according to the common core standard system: Newsela, "Appendix B of Common Core Standard" (AppBCCS) corpus, and a Mixed corpus, made by the union of the two.

Parallel to these three corpora, we implemented also two other datasets not based on the Common Core Standard, namely WeeBit and OneStopEnglish; these datasets will be used as a comparison to verify the correctness of the approach with diverse classification systems.

In the following sections, we are going to cover each corpus individually.

3.2.1 Newsela

Newsela is a corpus provided by Newsela Inc.[1], an Instructional Content Platform that provides content with integrated assessment, insights, and quizzes for students practicing reading capability according to the Common Core Standard.

The Corpus consists of 9564 English text samples, composed by 1910 English articles spanning various fields, and up to six simplified versions for each piece. The articles are classified in 11 levels corresponding to grades of the American school system (from "02" to "12"); however. for consistency with the other datasets, we grouped them into the following levels: "02-03", "04-05", "06-07-08", "09-10" and "11-12".

The composition of the Dataset provides an insight into the nature of the text contained; hence, we can expect convergence on the variety of topics and words used. Considering the possible relevant amount of lemmas that the text samples shares, it might prove out to be problematic to compute the Lexical Complexity. For this reason, we executed a comparison to understand the relevance of the overlapping of lemmas and the number of unique lemmas per level. The results can be seen in Table 3.1 and Table 3.2, calculated without considering all stop words and proper names.

[1] https://newsela.com

Table 3.1: Table showing the number of lemma and unique lemma in the various complexity levels of Newsela corpus

Comp. Level	Type	Token	Unique Type	Unique Token	Text Samples
02-03	4923	113629	44	47	724
04-05	12290	777968	259	339	2911
06-07-08	18037	1158677	343	424	3305
09-10	13476	288858	100	106	770
11-12	22155	818505	3814	5757	1854

Table 3.1, shows information concerning lemmas, unique lemmas, and text samples. In particular, lemmas are classified in two distinctive ways: the number of types and number of tokens. The "type" takes into account only the presence of a certain lemma, while the "token" considers the total occurrences of lemmas.

For example the sentence:

Mary likes cats, Mike instead hates all animals, including cats.

has the following lemmas (after removing proper names and stop words):

[like, cat, instead, hate, animal, include, cat]

for a total of 7 lemmas. For this sentence, "type" is 6, while "token" is 7, the reason being that the lemma "cat" appears two times, so it is counted only once while considering the number of types.

Looking more in details at Table 3.1, we can notice multiple elements:

- The variety of lemma per class (column "Type"), increases at the increment of the complexity level; except for the class "09-10" (Probably associated with a lower amount of data; see column "text samples").
- The increment in the variety of lemma is not strictly dependent by the total number of occurrences of the lemmas (column "Token") or the number of text samples (column "text samples"); in fact, class "09-10" present more "Type" than class "04-05", despite having lower values for both"Token" and "text samples".
- Simpler complexity levels present more shared lemmas with a higher complexity level and not vice-versa, suggesting that higher complexity levels use a wide range of lemmas that also incorporate the lower levels ones. (see column "Unique Type").
- The unique lemmas (column "Unique Type") do not represent the majority of lemma variety if we compare the values of "Unique Type" and "Type", the unique lemmas count in average for less then 10% of the total variety, however just a small increase in the variety seems to imply an increase in the complexity.
- Lastly, unique lemmas (column "Unique Type") are not very frequent, in fact by comparing "Unique Type" and "Unique Token", it is evident that every lemma is counted in average less than two times. This behavior suggests that the increment of complexity is not strictly dependent on the frequency of usage of the new lemmas, but only by the variety introduced.

Table 3.2: Triangular table showing the lemmas (type) that are common to two complexity levels in Newsela corpus

Complexity Level	02-03	04-05	06-07-08	09-10	11-12
02-03	**4923**	-	-	-	-
04-05	4835	**12290**	-	-	-
06-07-08	4841	11921	**18037**	-	-
09-10	4389	9434	12402	**13476**	-
11-12	4811	11708	17312	13197	**22155**

While in Table 3.1 the focus was posed on the number of lemmas and the unique lemmas associated with every class, in Table 3.2 it is highlighted the overlapping of lemmas among the various levels.

Table 3.2 is a triangular matrix, for this reason, it is proposed only in its lower triangle form. If needed, it can be easily read as a normal table, by examining first the row until the value on the main diagonal and then considering the column.

Looking at the table, as hypothesized while analyzing Table 3.1, is evident that the highest correlation in term of used lemmas is present with the levels representing a higher complexity compared to the current one. If we examine each value on the main diagonal (the values highlighted in bold), all the values that are on the associated row, will be lower compared to all the ones in the associated column.

3.2.2 Appendix B of Common Core Standard

Appendix B of Common Core Standard corpus (AppBCCS) consists of 168 texts given as sample texts to help the teachers define the level of knowledge required at each grade of the American school system. In particular, texts are grouped into the following levels: “02-03”, “04-05”, “06-07-08”, “09-10” and “11-12”.

It was presented in Appendix B of the “English Language Arts Standards of the Common Core State Standards”. [49]

Only a subset of the original 168 documents was available in digital format; thus, to improve the corpus, we tried to locate new books belonging to series from authors we found into the corpus. The dataset was then composed of 165 documents and further divided into chapters, treated separately, for a total of 2349 text samples.

AppBCCS, contrary to Newsela, presents a high variety of genre; all the text samples can be grouped in 5 main categories: Informational Texts (IT), Stories (S), Informational Texts: History and Social Studies (ITHSS), Informational Texts: Science, Mathematics, and Technical Subjects (ITSMT), Informational Texts: English Language Arts (ITELA), and Drama (D). To understand the relevance of

each genre, we can take a look at Figure 3.3.

In Figure 3.3a, where the number of chapters (equivalent to the number of single text samples) is organized in the various complexity levels and genre, it is immediately evident that the distribution of data is not equal. For example, the percentage of "Stories" tends to decrease at the increase of the complexity level.

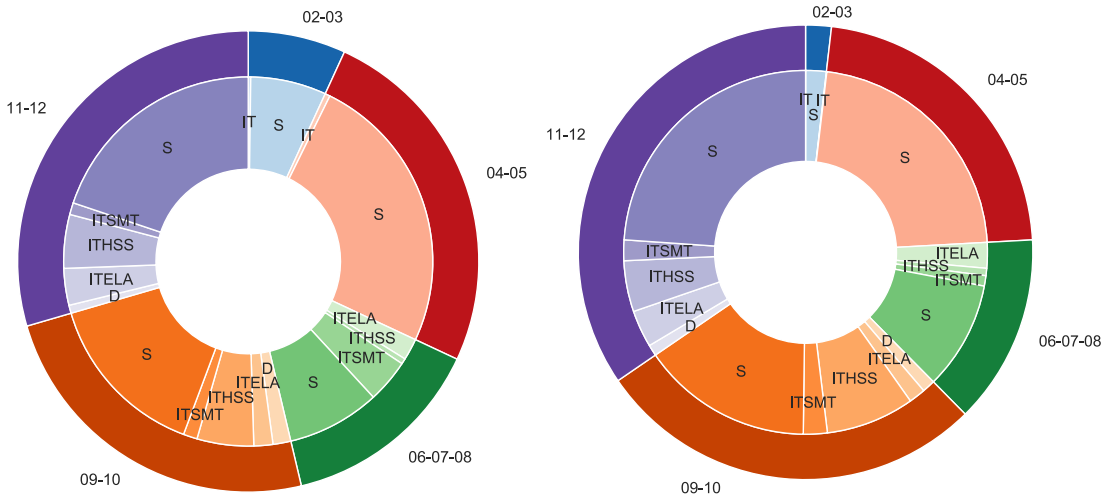
Looking in more detail is possible to see that the diversification of chapters tends to grow by the increase of the level of complexity. For example, the first two classes present only general "Informational Text" and "Story"; while, the higher complexity ones, introduce major genre diversification.

This phenomenon is probably caused by the limited amount of works available for the younger audience compared to the greater market for teenagers and young adults. Prove of this event is also the appearance of"Drama" only in the two most complex levels.

After identifying the imbalance that characterizes the dataset based on genre and number of text samples, we need to execute a deeper analysis to verify if the inequality is also associated with the structure and content of the chapters. For this reason, we propose Figure 3.3b, an analogous representation of the previous figure, but focused on the number of tokens.

By looking at the picture and confronting the proportions with the previous image, it is evident that the inequality is translated also at the token level, however, the discrepancy on the number of text samples does not necessarily imply the token's one, as we can see, for example, in class "06-07-08", and "09-10".

For more detail about the composition of AppBCCS, it is possible to check Appendix 6.2, in which we present a description of the relevance of every author in



(a) Chapters **(b)** Tokens

Figure 3.3: Chapters and tokens distribution in AppBCCS corpus

Table 3.3: Table showing the number of lemma and unique lemma in the various complexity levels in AppBCCS corpus

Comp. Level	Type	Token	Unique Type	Unique Token	Text Samples
02-03	4032	53177	70	174	185
04-05	14446	636209	1535	3984	669
06-07-08	25189	391740	10378	12713	343
09-10	24956	811073	5706	11498	494
11-12	26293	965600	6805	13490	658

Table 3.4: Triangular table showing the lemmas (type) that are common to two complexity levels in AppBCCS corpus

Complexity Level	02-03	04-05	06-07-08	09-10	11-12
02-03	**4032**	-	-	-	-
04-05	3654	**14446**	-	-	-
06-07-08	3704	10254	**25189**	-	-
09-10	3846	11676	13135	**24956**	-
11-12	3821	11632	13363	17504	**26293**

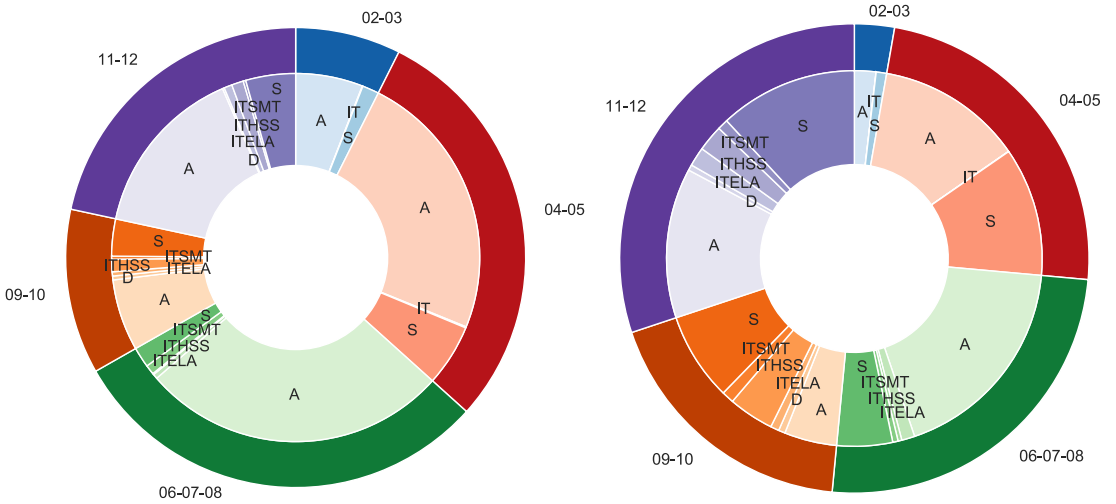
our version of the dataset, considering chapters, unique lemmas, and words.

Lastly, for coherence with the analysis proposed for the Newsela corpus, we are going to present the same results concerning the diversification of lemmas, among the complexity levels, in Table 3.3 and Table 3.4.

Doing a comparison to what was discovered while considering Newsela, it is visible that some properties are maintained. Examples are, the non-dependency between the variety of lemmas (column “Type”) and the total number of lemmas (column “Token” or text samples (column “text samples”) and the contributions provided by the unique lemmas in the count of all occurrences.

It is, instead, violated the first mentioned property, in fact, for class “09-10”, even if the number of text samples (column “text samples”) is higher then the one for level “06-07-08”, the quantity of lemmas (column “Type”) is lower; this phenomenon is probably the consequence of an anomalous high diversification of lemmas in class “06-07-08”. Consequence and probably proof of the existence of this effect is also the irregularity in the variety of unique lemmas for class “06-07-08”.

Considering Table 3.4, instead, it is easily seen that the previously identified property remains valid.



(a) Chapters **(b)** Tokens

Figure 3.4: Chapters and tokens distribution in Mixed corpus

3.2.3 Mixed

Mixed is the third dataset we implemented and is the union of Newsela and AppBCCS corpora.

Being the union of the two datasets, Mixed also carries the problems mentioned in the two previous sections; we are going to verify and analyze these problems using a similar approach.

Firstly, we can look at the pie charts that illustrate the proportion of chapters and tokens by dividing them according to complexity levels and genres (Figure 3.4). The graphs prove the hereditaries of the problems identified while analyzing AppBCCS. In this case, such problems are worsened due to the difference in size between Newsela and AppBCCS, with Newsela article(A), begin the dominant section in every complexity level.

Diverse and interesting, is instead the condition for the tokens. It is possible to notice that the relationship among Articles and AppBCCS writings change

Table 3.5: Table showing the number of lemma and unique lemma in the various complexity levels in Mixed corpus

Comp. Level	Type	Token	Unique Type	Unique Token	Text Samples
02-03	6733	166806	85	125	909
04-05	19138	1414177	1402	3084	3580
06-07-08	32922	1550417	10228	12366	3648
09-10	27955	1099931	4523	8530	1264
11-12	33662	1784105	7406	13544	2512

Table 3.6: Triangular table showing the lemmas (type) that are common to two complexity levels in Mixed corpus

Complexity Level	02-03	04-05	06-07-08	09-10	11-12
02-03	**6733**	-	-	-	-
04-05	6422	**19138**	-	-	-
06-07-08	6496	16160	**32922**	-	-
09-10	6359	15522	18989	**27955**	-
11-12	6533	16865	21632	22372	**33662**

drastically in favor of the second, probably suggesting that even if AppBCCS has fewer text samples then Newsela, the latter has shorter and less diversified texts compared to the former.

Secondly, to present a complete analysis of the corpus, Tables 3.5 and 3.6, display the results of a simple study on the diversification of lemmas.

Studying Table 3.5 it is obvious that the same problems introduced by AppBCCS are directly inherited by Mixed corpus, with class "06-07-08" re-proposing an anomaly behavior compared to the supposed standard one.

Lastly, looking at Table 3.6, we can discern that the correlation level is coherent to what was noticed for the two previous datasets, where the higher correlation on shared lemmas is with higher complexity levels then the current one, and not with lowers one.

3.2.4 WeeBit

WeeBit is a corpus composed of five complexity levels, the former three belonging to the Weekly Reader Magazine for children, and the latter two associated with the BiteSize learning platform of BBC. The dataset consists of 9709 text samples, with most of them (around 70% of the total) belonging to the last complexity level.

WeeBit is also the first dataset that we implemented that uses a different

Table 3.7: Conversion table from WeeBit class system to Common Core Standard one

WeeBit levels	Age of reference	CCS Levels
WRLevel2	7-8	02-03
WRLevel3	8-9	03-04
WRLevel4	9-10	04-05
BitKS3	11-14	06-07-08
BitGCSE	14-16	09-11

Table 3.8: Table showing the number of lemma and unique lemma in the various complexity levels in WeeBit corpus

Comp. Level	Type	Token	Unique Type	Unique Token	Text Samples
WRLevel2	3252	40860	301	542	599
WRLevel3	4708	63471	592	885	779
WRLevel4	6669	101442	1076	1683	802
BitKS3	4340	71172	409	733	646
BitGCSE	15237	1044269	8686	72743	6883

Table 3.9: Triangular table showing the lemmas (type) that are common to two complexity levels in WeeBit corpus

Complexity Level	WRLevel2	WRLevel3	WRLevel4	BitKS3	BitGCSE
WRLevel2	**3252**	-	-	-	-
WRLevel3	2440	**4708**	-	-	-
WRLevel4	2616	3653	**6669**	-	-
BitKS3	1783	2239	2748	**4340**	-
BitGCSE	2593	3573	5030	3839	**15237**

classification system compared to the grade division proposed by the Common Core Standard. However, given the division in years of the target reader, we can create a sort of conversion from the used classification system to the common core one. Such translation is proposed in Table 3.7.

Since a translation exists, it would be possible to treat every level as the corresponding CCS one. Unfortunately, considering the overlapping introduced by the class "WRLevel3" and the different categorization for level "BitGCSE", it would only cause noise during the analysis. For this reason, we decided to maintain the originally proposed division.

Following the line of analysis proposed for the previous dataset, also with this corpus, we are going to show the relationship, in terms of lemmas, among the complexity levels. Such analysis is reported using Tables 3.8 and 3.9.

Looking at Table 3.8, it is evident that a discrepancy is present among the levels. For example, considering level "WRLevel4" and "BitKS3", every metric gives smaller results, with the higher complexity level showing less variety and quantity of lemmas. The problem persists in the following level "BitGCSE", in which the numbers are indeed better, but only because the last level has a number of text samples of one magnitude higher than the other classes. It is highly probable that, if the number of text samples was similar to the other classes, the returned

value would be inferior to "WRLevel4".

We believe that this phenomenon might be the result of the different nature of the two magazines from which the articles are taken.

Since the levels belong to different magazines, we can expect that the classification methodology implemented by the two companies is not uniform. Furthermore, Weekly reader is an American company that creates content for children in American English, while BBC is an English company, that creates content for children in UK English.

The effects of the previous phenomenon are also reflected in the results obtained in Table 3.9, in which the problem of the two different variants of English is further highlighted from the lower number of shared lemmas between the levels belonging to Weekly Reader and BiteSize.

From the highlighted problems, we can expect that WeeBit will not perform well with our approach, at least for the part concerning the Lexical Complexity, given the intrinsic difference between the two variants of English.

3.2.5 OneStopEnglish

OneStopEnglish is the last dataset we are going to use. In this case, as for WeeBit, the classification system is different compared to the Common Core Standard. However, differently from WeeBit, the dataset is created similarly to Newsela, with 189 articles that are proposed in 3 different levels of complexity: Elementary, Intermediate, and Advanced.

Table 3.10: Table showing the number of lemma and unique lemma in the various complexity levels in OneStopEnglish corpus

Comp. Level	Type	Token	Unique Type	Unique Token	Text Samples
Elementary	4435	42855	81	93	189
Intermediate	6486	56582	219	1181	189
Advanced	8548	68777	2272	3031	189

Table 3.11: Triangular table showing the lemmas (type) that are common to two complexity levels in OneStopEnglish corpus

Complexity Level	Elementary	Intermediate	Advanced
Elementary	**4435**	-	-
Intermediate	4296	**6486**	-
Advanced	4305	6218	**8548**

Unfortunately, no information concerning the target age of such articles is provided. Hence it is not possible to create a direct translation with CCS classification rules, as we did for WeeBit.

Following a similar approach with the previous datasets, we proposed an analysis of the nature of the lemmas included in the corpus. We expect to see results in line with what we noticed in Newsela, given the similar constitution of the two corpora.

Table 3.10 shows that the dataset was built in a very accurate way. In fact, as a consequence of the equilibrate distribution of text samples in the corpora, the number of types and tokens grows almost equivalently among the levels.

In line with the expectation are also the results reported in Table 3.11, in which, although in a limited way, given the low number of levels, it is possible to notice a similar behavior with the previous datasets.

3.2.6 Remarks

In this chapter, we presented the five corpora that will be used in this thesis, together with some simple analysis about their nature.

The five corpora will be used both while computing Lexical Complexity and while computing Syntactic Complexity.

To be more precise, during the Lexical Complexity section, we will utilize Mixed as a base to build the vocabulary and then use all the other corpora as quality metrics when studying the correlation with the identified low-level metrics. Lastly, after describing how to generate the scale that assigns semantic to the score, we will produce an example based on Newsela.

During the Syntactic Complexity section, instead, we will use both Mixed and Newsela to train the networks, obtaining a total of 4 different models. Then each model will be tested using all the available datasets. Lastly, after describing the process to generate the scaling system, we will display an example based on Newsela.

Chapter 4

Lexical Approach

4.1 Model

In this section, we present our approach based on the log-likelihood ratio test, for associating Lexical Complexity scores to documents. Then, we will introduce our algorithm and explain its core functions.

4.1.1 Log-likelihood Ratio Test

The log-likelihood ratio test is a centroid-based, unsupervised content selection technique introduced by [50]. This technique aims to find the words that characterize a document and, if these words appear in a sentence, assign a weight to them.

The log-likelihood ratio is computed as $D = -2\log(\lambda(w_i))$ and define the importance of a word type w_i in the current document compared to the rest of the collection. D is referred to as *discrepancy*. This measure reflects the differences of the observed word frequencies in the current document, compared to the values that we would expect to see if the frequencies of the words were the same in the current document and the rest of the collection.

This implies that larger discrepancies reflect a higher difference between the word frequencies in the current document and the one in the others. To be more specific, D is defined as

$$D = 2\left[C_{doc}(w_i)\cdot\log\left(\frac{C_{doc}(w_i)}{E_{doc}(w_i)}\right) + C_{oth}(w_i)\cdot\log\left(\frac{C_{oth}(w_i)}{E_{oth}(w_i)}\right)\right] \tag{4.1}$$

where $E_{doc}[w_i]$ is the expected value of $C_{doc}(w_i)$ calculated as

$$E_{doc}[w_i] = \frac{N_{doc}}{N_{doc}+N_{oth}}\cdot(C_{doc}(w_i)+C_{oth}(w_i)) \tag{4.2}$$

and $E_{oth}[w_i]$ is the expected value of $C_{oth}(w_i)$ calculated as

$$E_{oth}[w_i] = \frac{N_{oth}}{N_{doc}+N_{oth}}\cdot(C_{doc}(w_i)+C_{oth}(w_i)) \tag{4.3}$$

in which, every variable can be defined as follow:

- $C_{doc}(w_i)$: number of occurrences of the word w_i in the current document
- $C_{oth}(w_i)$: number of occurrences of the word w_i in all the documents but the current one
- N_{doc}: number of tokens in the current document
- N_{oth}: number of tokens in all the documents but the current one

For huge corpora, $D \sim \chi^2(df = 1)$. Hence it is possible to consider the relevance of D according to a specific percentile. For example, if $D > 10.83$, w_i is significant for the current document, with at least 0.999 of significance level (99.9th percentile; $p < 0.001$).

To make this approach meaningful in the context of our goal, the variables are redefined as:

- $C_{doc}(w_i)$: number of occurrences of the word w_i in the current complexity level
- $C_{oth}(w_i)$: number of occurrences of the word w_i in all the complexity levels but the current one
- N_{doc}: number of tokens in the current complexity level
- N_{oth}: number of tokens in all the complexity levels but the current one

The retrieved discrepancy is then processed as follow:

1. Test the discrepancy against a 99.9th percentile
2. If the test fails for every complexity level, lower the threshold to 99th and try again.
3. If the test fails again for every complexity level, lower the threshold to 95th and try again.
4. If the test fails again for every complexity level, simply ignore the word.
5. If the test succeed assign the word to the lowest complexity level that passed the test.

By applying these simple steps, we can use D to associate every word to a specific complexity level, for which the word will be relevant.

4.1.2 Algorithm

In this section, we propose our algorithm, constituted of two main functions: Vocabulary Generation (Algorithm 1) and Score Computation (Algorithm 2).

Algorithm 1 describes the application of the log-likelihood ratio test and the assignment of a complexity level to every retrieved lemma; then, Algorithm 2 shows how to generate a score for a specific document. A complete explanation of the two algorithms is presented, respectively, in Section 4.1.2 and 4.1.2.

Algorithm 1: Vocabulary Generation

Algorithm 1: Vocabulary Generation

Data: Dataset *documents*, divided into complexity levels
Result: *lemmaScores*, a list of lemmas, each one associated with a complexity levels
begin
  $lv \leftarrow$ [“02-03”, “04-05”, “06-07-08”, “09-10”, “11-12”];
  $data \leftarrow$ PreProcessDataset(*documents*);
  for *lemma* ***in*** *data* **do**
    $countInLevels[lemma, *] \leftarrow$ CountInLevels(*lemma*, *lv*);
    $test[lemma] \leftarrow$ LogLikelihood($countInLevels[lemma, *]$, *data*);
    $cmplLevel[lemma] \leftarrow$ AssignComplexityLevel($test[lemma]$);
    $lemmaScores[lemma] \leftarrow$ AssignScore($cmplLevel[lemma]$);
  end
end

Algorithm 1 is composed of five core functions: *PreProcessDataset*, *CountInLevels*, *LogLikelihood*, *AssignComplexityLevel*, and *AssignScore*.

In the *PreProcessDataset* function, every text sample is parsed using spaCy, a free, open-source library for advanced NLP tasks in Python. Every text sample is hence represented in functions of the words that compose it, in particular, the function will consider only words that are not preemptively marked as stop words or proper nouns, and transform them in lemmas.

The standard stop-words list provided by spaCy was not fit for our task, for this reason, it has been customized, retaining only words that satisfy the log-likelihood ratio test (Section 4.1.1) for *every* complexity level. In other words, from our point of view, a stop-word is a word that is common to all the complexity levels and thus does not carry useful information. Finally, proper nouns are not considered because they do not carry intrinsic complexity (as they are just names) and, at run time,

Table 4.1: Table representing complexity levels and associated values

Complexity Level	"02-03"	"04-05"	"06-07-08"	"09-10"	"11-12"
Associated Value	0	0.25	0.50	0.75	1

they are likely to be outside of the vocabulary.

After the pre-processing phase, the algorithm calculates the number of instances of each lemma into each complexity level. (*CountInLevels*).

Then, *LogLikelihood*, as the name implies, is responsible for applying the log-likelihood ratio test as described in Section 4.1.1. This function represents the core of Algorithm 1, because it allows us to identify which word is characteristic of a certain complexity level. Once every lemma has been tested, the algorithm assigns a complexity level to each of them, if a lemma has not satisfied the test for any of the complexity levels then it is marked as not relevant (see Section 4.1.3 for details) and if a lemma satisfies the test for multiple levels, then the lowest one is considered (*AssignComplexityLevel.*)

Lastly, before returning the vocabulary, every complexity level is converted into a score, according to Table 4.1, which defines a normalized 0-1 scale, with equidistant values (*AssignScore.*)

Algorithm 2: Score Computation

Algorithm 2: Score Computation

```
Data: a vocabulary of lemmas with the associated complexity levels; a
      document to process
Result: Score s associated to the document
begin
    for sentence in document do
        data ← PreProcessSentence(sentence);
        for lemma in data do
            cmplLevel[sentence, lemma] ←
              AssignComplexityLevelS(lemma, vocabulary);
        end
    end
    s ← ComputeScore(cmplLevel);
end
```

Algorithm 2 is simpler than Algorithm 1 and it can be described by three functions: *PreProcessSentence*, *AssignComplexityLevelS* and *ComputeScore*.

Similarly to Algorithm 1, the first function (*PreProcessSentence*) is responsible of the pre-processing stage; the analysis, however, is executed sentence-wise and not document-wise. This means that the occurrences of lemmas are considered only in terms of a single sentence. The pre-processing is then followed by *AssignComplexityLevelS*, the function responsible for assigning to every lemma of the sentence its complexity level according to the provided vocabulary.

Finally, the function *ComputeScore* gives a score s to the document based on a weighted mean of the maximum complexity level assigned to a word in a sentence i, multiplied by the sentence length, as

$$s = \frac{\sum_i^n \max(V_i) \cdot l_i}{\sum_i^n l_i} \tag{4.4}$$

where n is the number of sentences in the document, l_i is the original length of sentence i, including stop words, proper names, and numbers; V_i represents the list of associated values to the processed words in the sentence i (assigned by *AssignComplexityLevelS*), excluding stop words, proper names, numbers, and Out Of Vocabulary (OOV) words. V_i can be defined as follow:

$$V_i = [v_{1,i},\ \ldots,\ v_{j,i},\ \ldots,\ v_{m_i,i}]; \quad j \in [1, m_i]; \quad v_{j,i} \in \{0, 0.25, 0.50, 0.75, 1\} \tag{4.5}$$

where $m_i \leq l_i$ is the number of considered words in the i-th sentence and $v_{j,i}$ is the complexity value assigned to the j-th parsed word of the i-th sentence; according to the scale presented in Table 4.1.

4.1.3 Handling OOV and non-relevant words

During the execution of both algorithms, it is possible to identify lemmas that are either not able to satisfy the Log-Likelihood Ratio Test (i.e., non-relevant words) or are not present in the vocabulary (i.e., OOV words). In these cases the possible approaches are four:

- Ignore these lemmas (Do not apply any automatic assignment)
- Assign the lemmas only at Score Computation step (OOV words)
- Assign the lemmas only at Vocabulary Generation step (non-relevant words)
- Assign the lemmas both at Vocabulary Generation and Score Computations steps (both non-relevant and OOV words)

The results of the application of these alternatives, in computing the score for the Mixed corpus (see Section 3.2), are shown in Figure 4.1, where a box represents the amount of data belonging to the Inter-Quartile Range (IQR: between

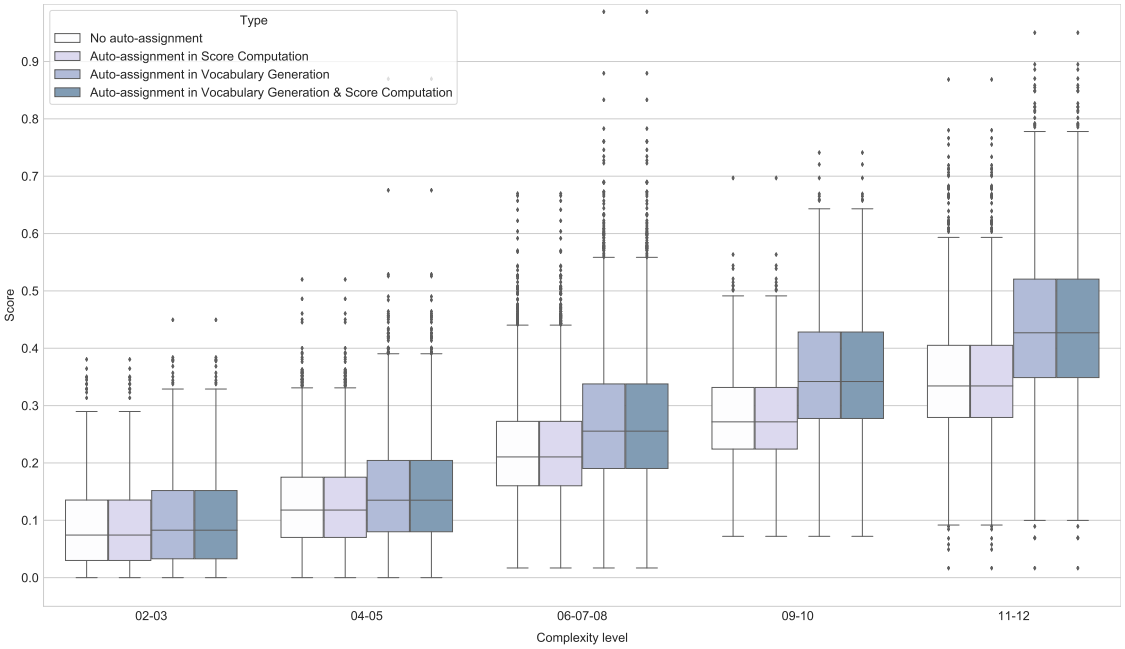


Figure 4.1: Score distribution in documents belonging to the five complexity levels (Mixed corpus); the graph highlights the differences among the four approaches for handling non-relevant and OOV words

the 25th quantile and the 75th quantile) while the line inside a box displays the median or 50th quantile; moreover, upper and lower whiskers represent, respectively, $25\text{th} - 1.5 \cdot IQR$ times and $75\text{th} + 1.5 \cdot IQR$; finally, the black dots represent samples outside the whiskers (outliers).

After testing the four alternatives, we noticed that assigning the complexity level to non-relevant words during the Vocabulary Generation step greatly increment the intra-level variance. If the assignment is, instead, done during the Score Computation phase, the increase is less relevant. However, the smallest variance is obtained just ignoring them, and so is our choice.

4.2 Experimental Results

In this section, we are going to explain how we implemented the algorithm described in Section 4.1.2, and the results that we obtained. Lastly, we will test the validity of the method by executing a comparison with a series of low-level indexes that identify variegate features of Lexical Complexity.

4.2.1 Vocabulary Generation

The first step in implementing the provided algorithm is to generate a vocabulary on which base the score computation, or in other terms, implement Algorithm 1.

When considering the best dataset for this scope, the decision must be driven by two factors: diversification of the lemmas and amount of documents. The reason for these constraints is quite simple; since we are going to use such dataset, as the base for the computation of the score, we want to minimize the number of lemmas that falls in the category of non-relevant words and out of vocabulary words (mentioned in Section 4.1.3).

Given these specifications, and using as support the data reported in Section 3.2 when presenting the datasets, it is evident that the most suited corpus is Mixed since it has the highest number of documents and the highest variety in term of genres (See Table 3.5 and Figure 3.4).

It is relevant to notice that during the execution of Algorithm 1, the preprocessing phase is responsible for identifying the lemmas associated with every document. As mentioned in Section 1, while performing this operation, stop words and proper nouns are discarded. In particular, the list of stop words used is not the one automatically imported by spaCy, but a customized version. This new list is obtained by executing a variant of Algorithm 1, here presented as Algorithm 3, in which no stop words are removed at the beginning if not for the basic pronouns and the auxiliary verbs "to be" and "to have".

Algorithm 3: Vocabulary Generation

Data: Dataset *documents*, divided into complexity levels, and original *stopWordsList* from spaCy

Result: *stopWords*, a list of lemmas identifying the new stop words list that will be used in Algorithm 1

begin

```
lv ← ["02-03", "04-05", "06-07-08", "09-10", "11-12"];
data ← PreProcessDataset(documents);
for lemma in data do
    countInLevels[lemma, ∗] ← CountInLevels(lemma, lv);
    test[lemma] ← LogLikelihood(countInLevels[lemma, ∗], data);
end
lemmas ← FindAllComplexity(test);
stopWords ← MaintainOnly(lemmas, stopWordsList);
```

end

The first part of the algorithm is equivalent to the original Algorithm 1, however, instead of *AssignComplexityLevel*, a new function is introduced to return all the lemmas that satisfy the log-likelihood ratio test for every level of complexity (*FindAllComplexity*). The output is then provided to *MaintainOnly*, which from

all the retrieved lists, maintain only the lemmas that are also included in the original stop words list provided by spaCy. If this last condition was not implemented, then it might happen that the already low number of lemmas in the lowest level would be even smaller, since most of the lemmas are common to all the levels.

4.2.2 Score Generation

Once, we defined the core dataset and generated the vocabulary, we can move to the second part of the process: Algorithm 2. The implementation is quite straightforward and the final result, computed for every dataset is presented in Figures 4.2 - 4.6.

For every dataset, we present the results both in the form of boxplot (top), and distribution (bottom); These plots provide information concerning variance, density, and overlapping present between each class. It is relevant to notice, however, that due to the presence of the lower margin in 0, the representations of the lower levels of the distributions are misleading. Some of the curves are represented below 0, but clearly, this is just a consequence of the nature of the graph itself. Figure 4.7 will provides a more precise representation (for Newsela),

By looking at the mentioned figures, it is evident how some datasets behave properly, showing an increase of the complexity with the increment of the marked level in the dataset, and distributions close to a Normal, while some perform very poorly. We will then present an analysis for each image, also motivating the anomalies that might be present.

Newsela

The first analyzed dataset is Newsela (Figure 4.2), this dataset is characterized by a good level of response from the algorithm, in fact:

- The distributions are overlapping but allowing every complexity level to be dominant in a particular range.
- The distributions have a median that increases at the increment of the complexity levels.
- The right tail has a softer slope compared to the left one, possibly suggesting that the data density is higher in the last quartile compared to the first one.
- The variance among the different complexity levels appears to be relatively stable, suggesting a good distribution of the data.

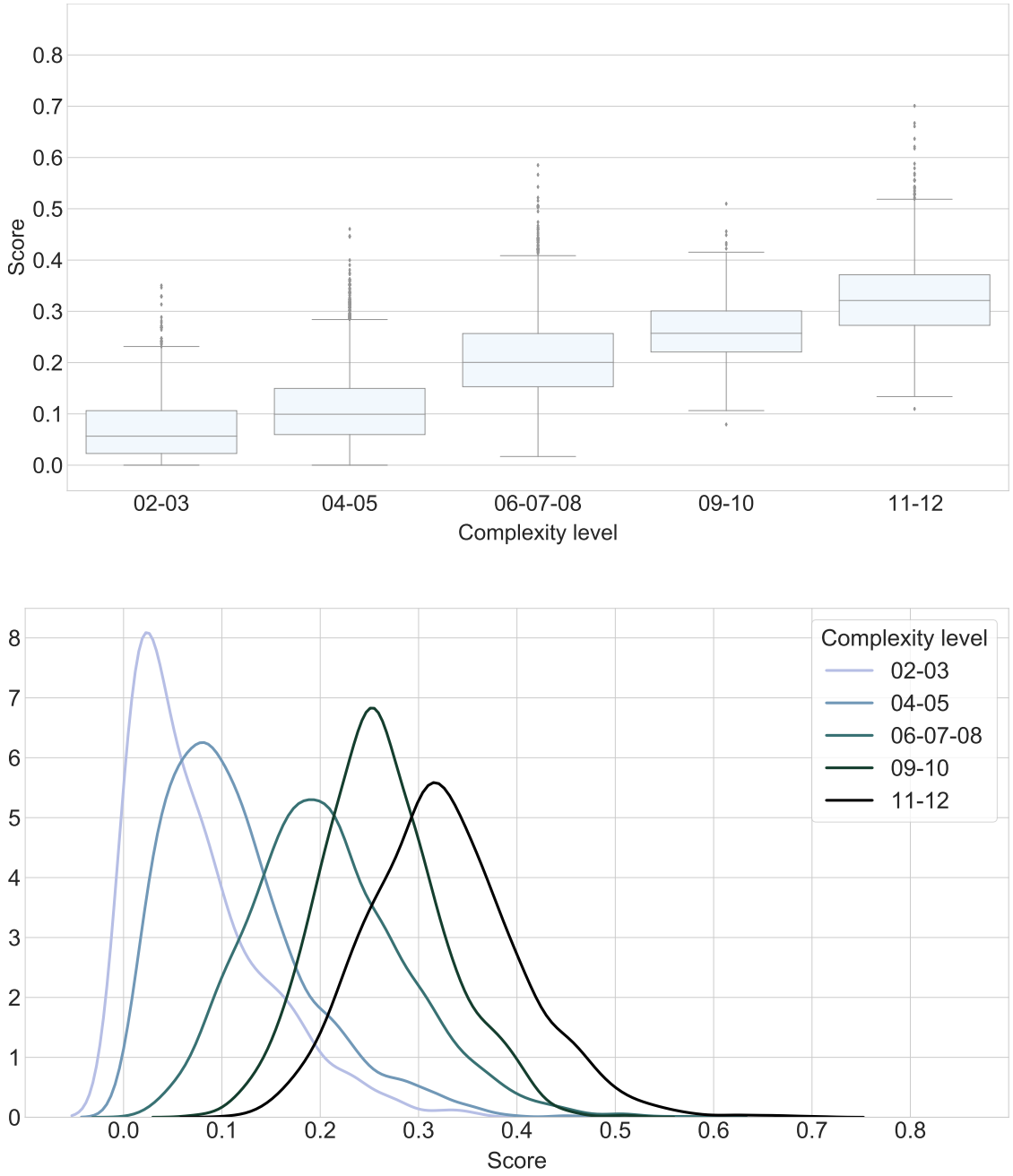


Figure 4.2: Box plot and distribution for the Newsela corpus

It is relevant to remember that since the dataset used for the vocabulary is a super-set of this corpus, we should expect that the algorithm encounter few if not none OOV words during the scoring phase.

AppBCCS

The second proposed dataset is AppBCCS (Figure 4.3), this corpus, in contrast with the previous one, presents a behavior extremely unstable, in particular:

- The curves display a high degree of overlapping, specifically the two central complexity levels ("06-07-08" and "09-10").
- The median seems to be in general increasing but an anomaly is present with the two levels mentioned above, with the levels showing an inverted position in the ranking.
- Except for the lower levels, the right tail is steeper than the left one, suggesting that all the document scores tend to converge on a central value, instead of showing a net distinction among classes.
- The variance is highly unstable, with the highest complexity level ("11-12") having a variance almost 3 times the smallest one.

This behavior is indeed strange since also AppBCCS is a sub-set of the corpus used while generating the vocabulary; in our opinion, the problem with AppBCCS is two-fold. Firstly, most of the dataset is constituted by books; unfortunately, assigning a complexity level to a whole book does not take into account that complexity can vary in different chapters. Secondly, most of the documents are novels, in which the complexity is strongly dependent on the writing style of the author. Both these factors negatively affect the algorithm, that treats every chapter independently and in which only the Lexical Complexity is computed.

Mixed

Now moving to the third dataset, we can observe the resulting score on Mixed corpus (Figure 4.4); as expected from this dataset, since it is the same dataset on used to generate the vocabulary, the outcome is considerably good. In particular, we can recognize behavior similar to the one manifested in Newsela, with the only difference being that the variance among the complexity levels is higher. For example, the level "06-07-08" contains text samples with a score of 0, while this was not happening within Newsela.

The increase in the variance, in particular, compared to Newsela, is probably a consequence of the addiction of the AppBCCS corpus.

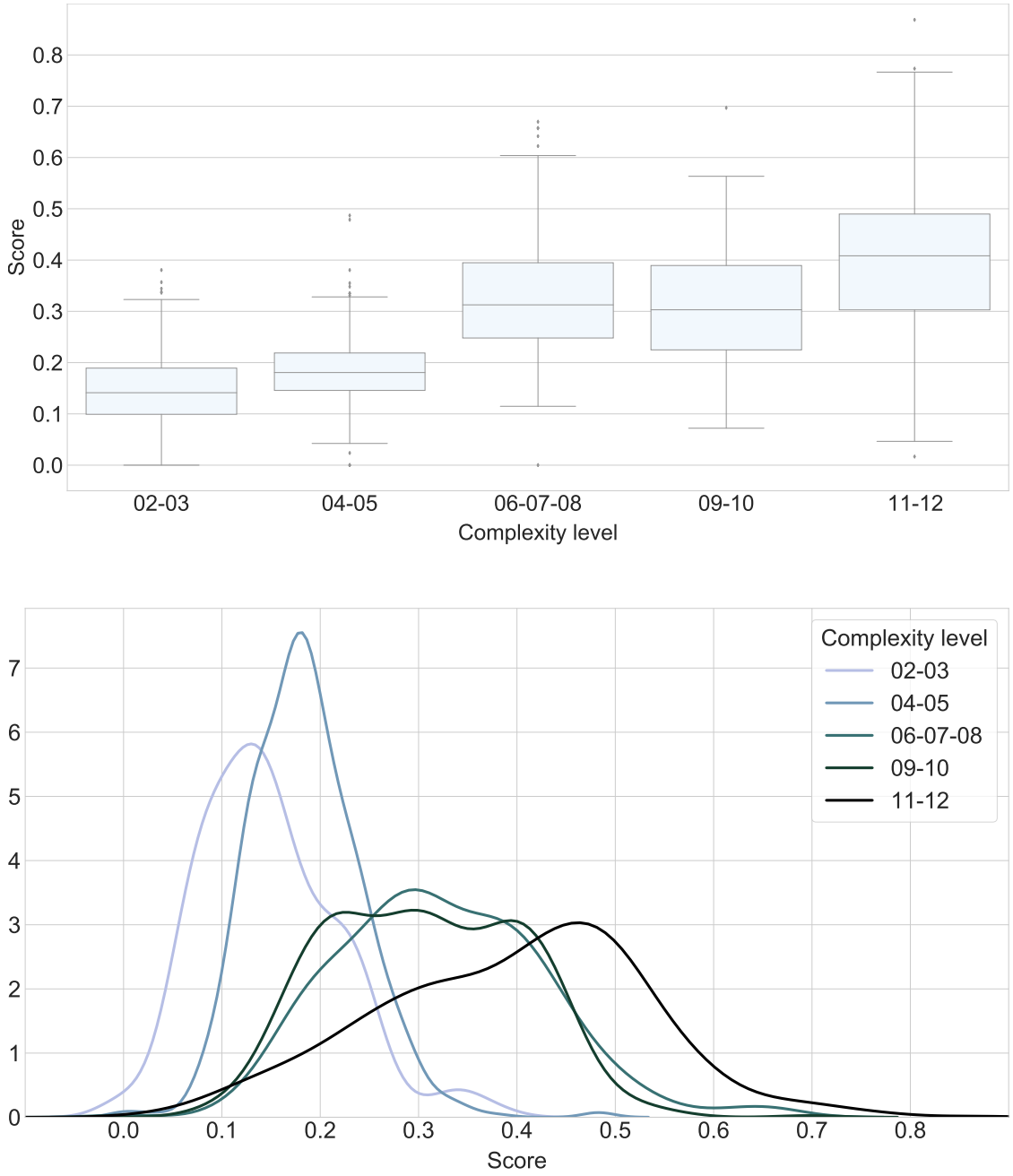


Figure 4.3: Box plot and distribution for the AppBCCS corpus

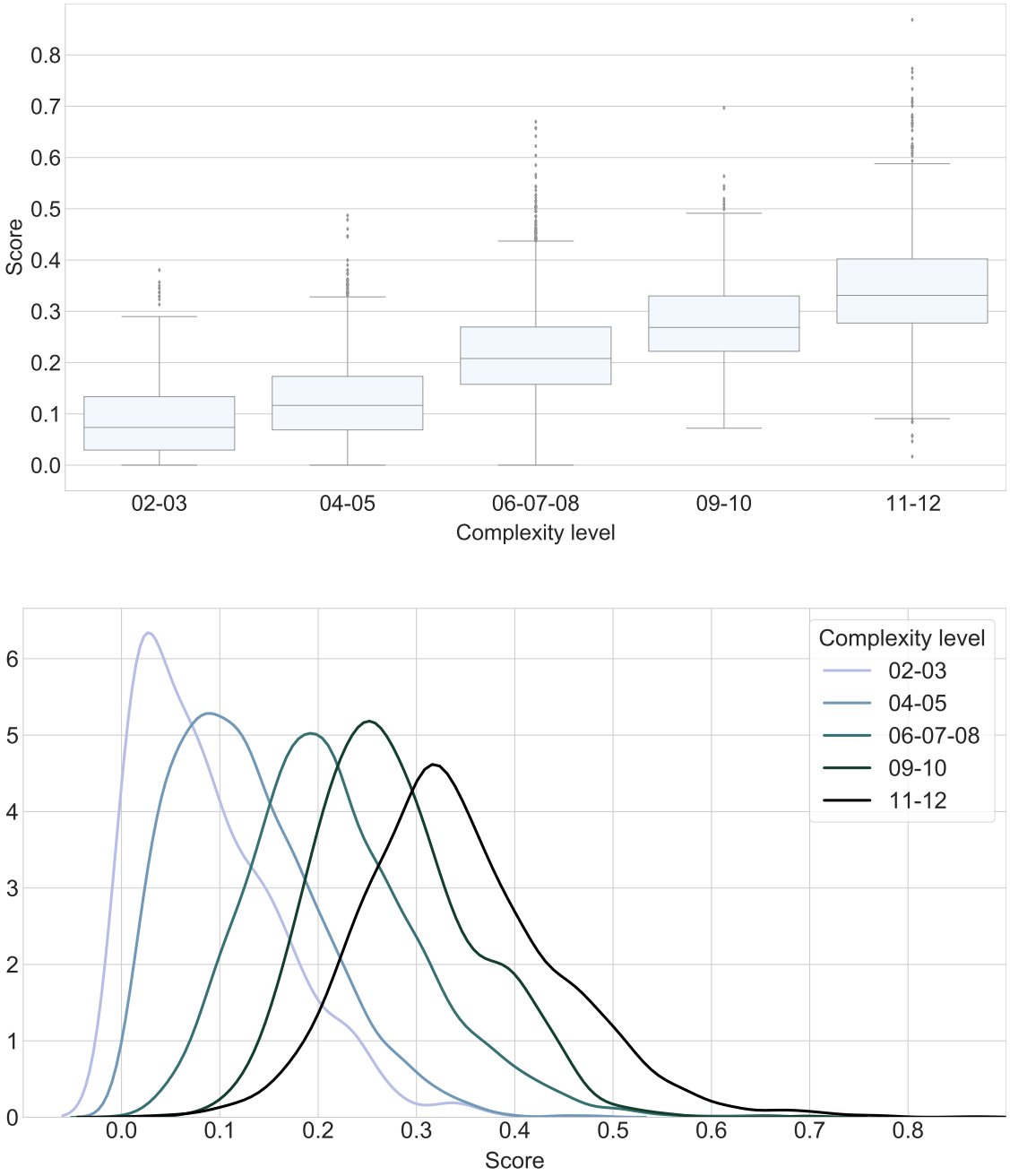


Figure 4.4: Box plot and distribution for the Mixed corpus

WeeBit

The fourth dataset, WeeBit (Figure 4.5) is the first corpus to be proposed that is non related in any way to the corpora used while generating the vocabulary; furthermore, it is also the first corpus to be characterized by a different classification system.

The outcome is terrible, with the scoring result being very unstable, generating peeks of complexity and strong overlapping on some levels.

This problem manifest within the couple "WRLevel2", "WRLevel3", and "BitKS3", "BitGCSE". In the former case, we have almost a complete overlap of the distributions, suggesting that our algorithm identifies the text samples included in these two classes as identical; while in the latter, the level "BitKS3" shows a complexity higher compared to "BitGCSE".

We believe that the problem is intrinsic to the text samples that compose the dataset; such texts are the result of a scraping activity from magazines for children, containing in most cases images and exercises for the young learner. These exercises often consist of "filling the gap" and question answering activities. Such "noisy" documents negatively affect the performances of Algorithms 1 and 2, implemented for pure text.

Furthermore, the corpus is composed of text samples written in both British English and American English. While this condition might seem irrelevant, in a context in which only the lemmas are analyzed it becomes predominant. The two variants of English manifest enough discrepancies in the glossary to confuse our algorithm based on an American English vocabulary.

OneStopEnglish

The last presented dataset is OneStopEnglish (Figure 4.6), also this dataset, as WeeBit is completely uncorrelated to the corpora used to generate the vocabulary, and it is organized in a different classification system.

However, contrary to WeeBit, the result is in line with what was identified for Newsela and Mixed corpora, suggesting that the algorithm is indeed working and that is applicable also on corpus different from the one used in generating the vocabulary.

Comparing these results with the previous, we can see a similarity with the features identified for Mixed and Newsela, with slightly more overlapping among the complexity levels, in particular with the two highest ones. This phenomenon probably suggests that the text samples of the "Intermediate" and "Advanced" complexity level, share a great number of lemmas.

Lastly, it is interesting to notice, how both OneStopEnglish and Newsela present

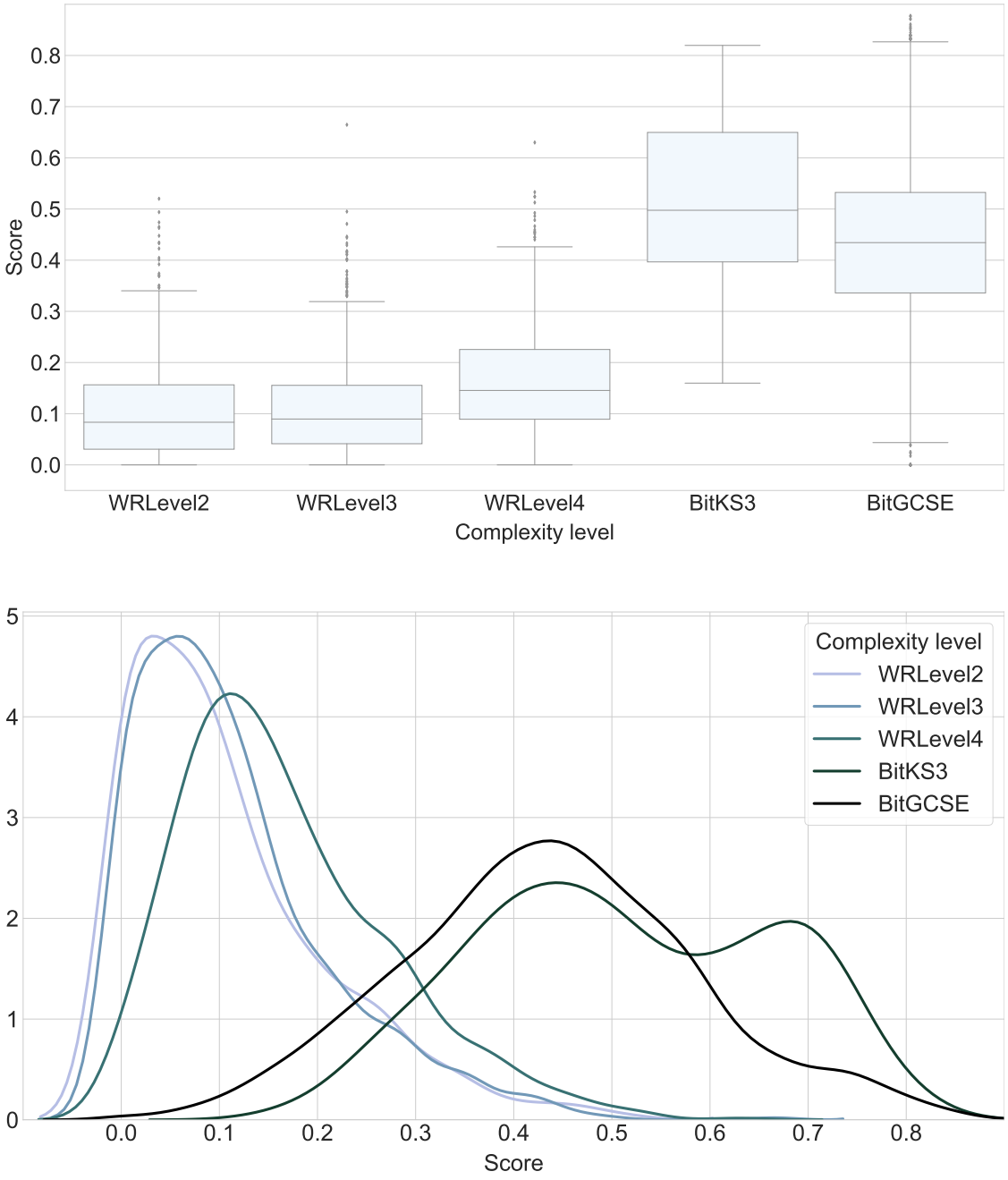


Figure 4.5: Box plot and distribution for the WeeBit corpus

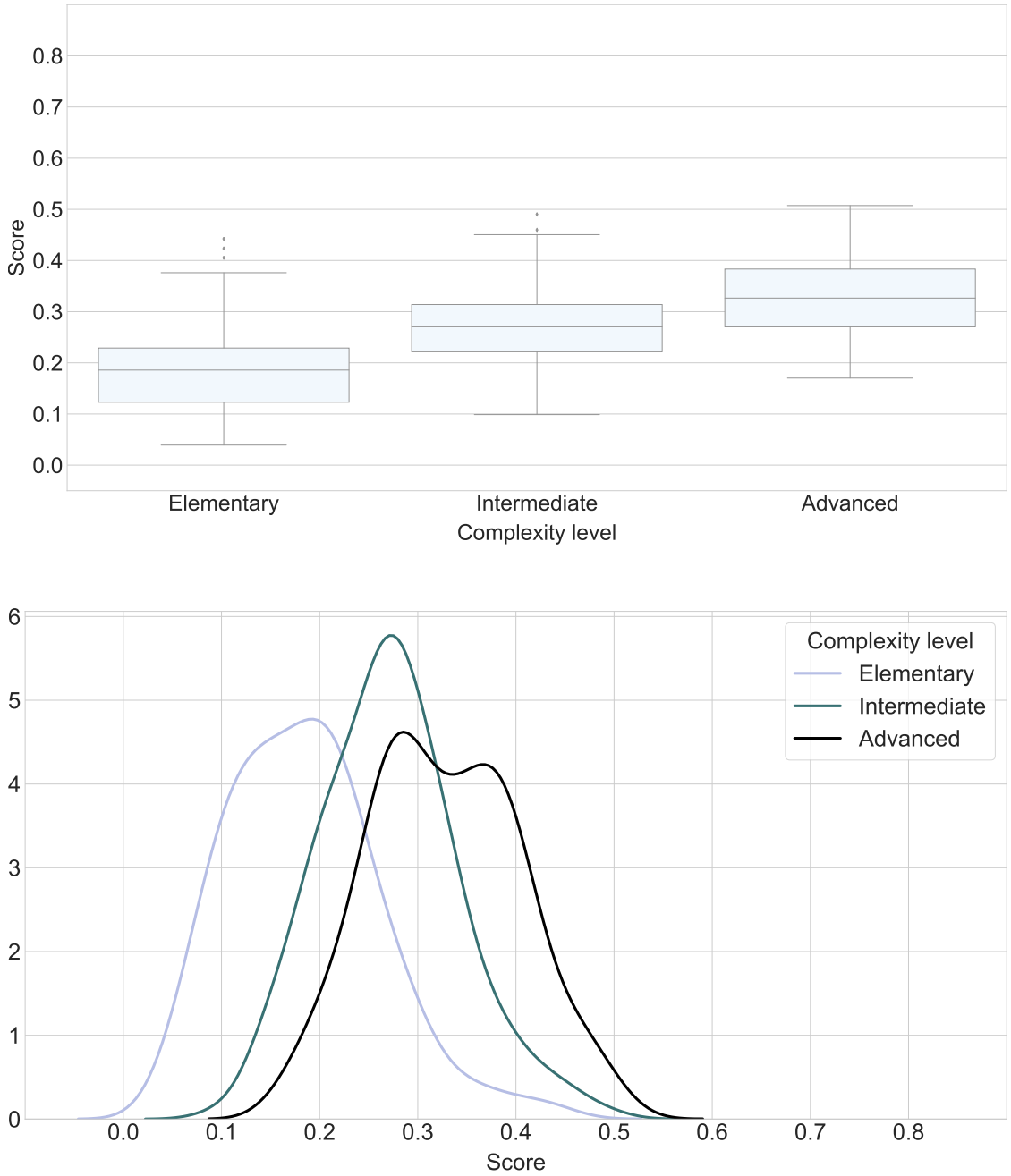


Figure 4.6: Box plot and distribution for the OneStopEnglish corpus

a similar higher and lower bound of the score, probably suggesting that using the Mixed corpus was a good choice in terms of generalization of the problem.

4.2.3 Score Validation

In the previous section, we described and analyzed the results obtained in applying the algorithm for all the proposed dataset, however, how can we be sure about the real quality of such scores?

Unfortunately, none of the above corpora classify texts considering *only* the Lexical Complexity; in fact, their classifications are based on the "generic complexity" of documents, mixing lexical, syntactic, and semantic aspects. Thus, we can not test out the effective validity of our approach in a classical style.

What we can do, instead, is to verify the correctness of our method by comparing its score with similar measures of Lexical Complexity. For this purpose, we implemented the strategy proposed by [23], an algorithm that calculates 25 distinct metrics covering several features of the Lexical Complexity.

Such metrics are arranged in 3 groups: Lexical Density (a metric that computes the ratio of the number of lexical words versus the total number of words in a text), Lexical Sophistication (5 metrics that analyze the proportion of advanced or unusual words in a document), and Lexical Variation (19 metrics that investigate the variety of words used in the document).

In particular, we evaluated the Pearson correlation coefficient and Spearman rank correlation between our score and such metrics. Table 4.2 displays the complete results, divided in category.

The meaning of such 25 metrics is briefly presented in the following list, together with the paper in which they were first introduced:

LD: Lexical Density [51]

CVS1: Corrected Verb Sophistication 1 [28]

LS1: Lexical Sophistication 1 [52, 53]

LS2: Lexical Sophistication 2 [54]

VS1: Verb Sophistication 1 [55]

VS2: Verb Sophistication 2 [56]

NDW: Number Different Words [57, 58]

NDWERZ: Number Different Words (Expected Random Z words) [59]

NDWESZ: Number Different Words (Expected Sequence Z words) [59]

Metric	AppBCCS		Newsela		Mixed		WeeBit		OneStopEnglish	
	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s
LD	0.30590	0.29985	0.26341	0.26186	-0.03221	0.04706	-0.08448	-0.06915	0.31971	0.30224
CVS1	0.45847	0.53287	0.67387	0.70254	0.59531	0.68000	0.02737	0.02515	0.55966	0.56315
LS1	-0.27274	-0.25897	0.24605	0.23627	0.24651	0.25187	-0.07784	-0.11921	0.22452	0.21232
LS2	0.49217	0.55870	0.65985	0.67774	0.64558	0.67751	0.12978	0.12567	0.51990	0.51422
VS1	0.65781	0.68797	0.65806	0.67687	0.69198	0.70130	0.12951	0.14855	0.55553	0.55343
VS2	0.39128	0.53287	0.59687	0.70254	0.43509	0.68000	-0.02762	0.02510	0.51131	0.56314
NDW	0.20736	0.26743	0.60058	0.64596	0.39507	0.58678	-0.10485	-0.11174	0.47183	0.48705
NDWERZ	0.37363	0.37130	0.45042	0.45882	0.24549	0.28848	0.13272	0.10599	0.41760	0.40485
NDWESZ	0.37284	0.35818	0.47501	0.48843	0.25425	0.29736	0.01797	-0.00914	0.36662	0.35586
NDWZ	0.08660	0.08945	0.37310	0.38097	0.21378	0.25177	-0.06004	-0.05238	0.24814	0.24134
CTTR	0.44346	0.53034	0.64717	0.67369	0.60701	0.66262	-0.04980	-0.09195	0.53069	0.54153
LogTTR	0.45124	0.48070	0.47657	0.48439	0.19717	0.27178	0.04406	0.10697	0.43398	0.42653
MSTTR	0.41792	0.39560	0.54091	0.56507	0.28088	0.33427	-0.02166	-0.05545	0.44177	0.44087
RTTR	0.44346	0.53034	0.64717	0.67369	0.60701	0.66262	-0.04980	-0.09195	0.53069	0.54153
TTR	0.24270	0.26378	0.20043	0.19005	-0.04380	0.02124	0.04760	0.11885	0.28018	0.27270
UBER	0.62385	0.69179	0.61791	0.63817	0.55510	0.58642	-0.01896	-0.00757	0.51535	0.52346
CVV1	0.38282	0.45832	0.59228	0.61382	0.55899	0.59553	-0.18020	-0.21513	0.54911	0.55490
SVV1	0.39029	0.45832	0.58384	0.61382	0.51858	0.59553	-0.19113	-0.21511	0.53852	0.55490
VV1	0.41069	0.41348	0.37908	0.37540	0.22161	0.25017	-0.03190	0.02142	0.45287	0.44876
AdjV	0.37323	0.36241	0.27655	0.26650	0.30645	0.28189	0.33053	0.34592	0.08227	0.09639
AdvV	-0.24270	-0.29114	0.00266	0.00860	-0.00931	-0.01185	-0.22242	-0.21920	0.06753	0.06652
LV	0.37252	0.38114	0.38284	0.38373	0.28855	0.30966	0.14837	0.19941	0.21006	0.21173
ModV	-0.24270	-0.29114	0.00266	0.00860	-0.00931	-0.01185	-0.22242	-0.21920	0.06753	0.06652
NV	0.41600	0.43532	0.37688	0.37842	0.32220	0.33233	0.17512	0.22724	0.20265	0.20654
VV2	-0.07216	-0.04766	-0.09192	-0.09044	-0.04663	-0.06170	-0.14320	-0.11507	0.12812	0.13665

Table 4.2: Pearson correlation coefficient (ρ) and Spearman rank correlation (ρ_s) between our score and standard lexical metrics, per dataset

NDWZ: Number Different Words (first Z words) [60]

CTTR: Corrected Type Token Ratio [61]

LogTTR: Bilogarithmic Type Token Ratio [62]

MSTTR: Mean Segmental Type Token Ratio [63]

RTTR: Root Type Token Ratio [64]

TTR: Type Token Ratio [65]

UBER: UBER Index [66]

CVV1: Corrected Verb Variation 1 [28]

SVV1: Squared Verb Variation 1 [28]

VV1: Verb Variation 1 [55]

AdjV: Adjective Variation [67]

AdvV: Adverb Variation [67]

LV: Lexical Word Variation [68, 51, 53, 52]

ModV: Modifier Variation [67]

NV: Noun Variation [67]

VV2: Verb Variation 2 [67]

To understand if the obtained results are relevant we evaluated the correlation strength adopting Cohen's interpretation. Specifically, if it is lower then 0.10, then there is no association of any kind; a correlation between 0.10 and 0.30 depicts a small association; a correlation within 0.30 and 0.50 represents a medium association; correlation higher than 0.50 signifies a large association.

Tables 4.3 and 4.4 displays the results of this interpretation, in a more convenient and easy to read representation.

Table 4.3 exposes the summed results for all the presented metrics, highlighting for every corpus, both Pearson correlation coefficient and Spearman rank correlation, the number of metrics with a correlation level Large, Medium, Small, or None according to the methodology presented before.

By looking at the table, it is evident that the best performing dataset is Newsela, immediately followed by OneStopEnglish, and Mixed with most metrics showing either a large or medium degree of correlation. In the fourth position, we find AppBCCS in which most of the metrics have a medium association with our score; in

Table 4.3: Correlation strength total

Strength	AppBCCS		Newsela		Mixed		WeeBit		OneStopEnglish	
	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s
Large	2	7	11	11	8	10	0	0	9	9
Medium	16	10	7	7	5	4	1	1	8	8
Small	2	3	4	4	7	6	5	7	5	5
None	5	5	3	3	5	5	19	17	3	3

the last position, we have WeeBit with the highest number of metrics not correlated to the computed score.

These results are in line with what was visible from the boxplot and distribution graphs, suggesting that the algorithm is indeed correct and applicable to multiple datasets. Also, the slightly worse behavior of AppBCCS and the terrible of WeeBit were in line with the expectation and the problems highlighted before while analyzing Figures 4.3 and 4.5.

After seeing the general result for every dataset, it easy to notice that some metrics do not correlate at all for multiple datasets. For this reason, and also to analyze in deeper detail the differences among the datasets, we present Table 4.4, in which the metrics are regrouped in categories.

As mentioned before the 25 metrics cover multiple aspects of Lexical Complexity, mainly grouped in Lexical Density (1 metric), Lexical Sophistication (5 metrics) and Lexical Variation (19 metrics), such organization is proposed also in the table, but instead of providing a general result for Lexical Variation, we decided to divide the 19 metrics into 4 subcategories, namely Number of Different Words (4 metrics), Type/Token Ratio (6 metrics), Verb Density (3 metrics), and Lexical Word Diversity (6 metrics).

The first group is Lexical Density, a metric that computes the ratio of the number of lexical words versus the total number of words in a text. This metric has been reported as not strictly relevant when considering the complexity of a text [52, 51], conclusions that are further supported by our results, with both Mixed and Newsela not correlating with the score value.

The second group consists of Lexical Sophistication (or Lexical Rareness), as the name suggests, these metrics analyze the proportion of advanced or unusual words in a document. These metrics are based on a similar conception to the core idea of our approach, as a consequence, they perform well, with almost all the metrics presenting a large level of correlation.

From the third group on, the metrics belong to Lexical Variation (or Lexical Diversity), such measures reflect the variety of words used in the document. The

Table 4.4: Correlation strength divided per category

Strength	AppBCCS		Newsela		Mixed		WeeBit		OneStopEnglish	
	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s
Lexical Density (1 Metric)										
Large	0	0	0	0	0	0	0	0	0	0
Medium	1	0	0	0	0	0	0	0	1	1
Small	0	1	1	1	0	0	0	0	0	0
None	0	0	0	0	1	1	1	1	0	0
Lexical Sophistication (5 Metrics)										
Large	1	4	4	4	3	4	0	0	4	4
Medium	3	0	0	0	1	0	0	0	0	0
Small	0	0	1	1	1	1	2	2	1	1
None	1	1	0	0	0	0	3	3	0	0
Number of Different Words (4 Metrics)										
Large	0	0	1	1	0	1	0	0	0	0
Medium	2	2	3	3	2	1	0	0	4	4
Small	1	1	0	0	2	2	1	1	0	0
None	1	1	0	0	0	0	0	0	0	0
Type/Token Ratio (6 Metrics)										
Large	1	3	4	4	3	3	0	0	3	3
Medium	4	2	1	1	0	1	0	0	2	2
Small	1	1	1	1	2	1	0	2	1	1
None	0	0	0	0	1	1	6	4	0	0
Verb Density (3 Metrics)										
Large	0	0	2	2	2	2	0	0	2	2
Medium	3	3	1	1	0	0	0	0	1	1
Small	0	0	0	0	1	1	0	0	0	0
None	0	0	0	0	0	0	3	3	0	0
Lexical Word Diversity (6 Metrics)										
Large	0	0	0	0	0	0	0	0	0	0
Medium	3	3	2	2	2	2	1	1	0	0
Small	0	0	1	1	1	1	2	2	3	3
None	3	3	3	3	3	3	3	3	3	3

wider is the vocabulary used, the higher will be the score presented by these metrics, with each of the subcategories being self-explanatory.

The third group, Number of Different Words, presents an average behavior, with the metrics mainly showing a medium correlation with our score, except for Mixed corpus. The reason probably resides in the influence generated by AppBCCS on the corpus.

The fourth group, Type/Token Ratio, shows a better behavior than the previous section, with most of the datasets exhibiting a large degree of correlation.

Immediately after we have Verb Density, also in this case the performance is excellent, with most of the datasets returning a high association with our score.

Lastly, the final group is Lexical Word Diversity, this is the worst-performing group, with half of the metrics showing zero correlation with the returned score. It is interesting to notice, that the metrics scoring lower are the same for almost all the datasets, being "AdvV", "ModV", and "VV2", and that these metrics consists also in the majority of the non-correlation results returned in Table 4.3.

Summing up, our score seems to correlate with most of the low-level metrics, being able to "concentrate" in just one value the semantics carried by multiple measures.

4.3 Scale Generation

Any scoring system, to be usable, needs some semantics; in other words, we need a *scale*. In this section, we are going to propose a possible approach to generate it.

The first step in defining a scale is to select which dataset to use as a base. Among the best-performing ones (Newsela, Mixed, and OneStopEnglish), we decided to use Newsela, since it has a higher amount of levels compared to OneStopEnglish, and a lower intra-level variance compared to Mixed. Thus, in the rest of the section, data are referring to the Newsela corpus.

The second phase consists of the analysis of the distributions of scores. To make this process more robust, we removed outliers. Outliers were identified using an approach based on the IQR factor; however, to take into account the skewness of data that Figure 4.2 (bottom) displayed, we adopted the approach proposed in [69] based on the *medcouple* [70]; so, for a given level c, our system retains all the scores for which

$$s \in \begin{cases} [Q_1 - 1.5IQR \cdot e^{3MC},\ Q_3 + 1.5IQR \cdot e^{-4MC}]; & MC \geq 0 \\ [Q_1 - 1.5IQR \cdot e^{4MC},\ Q_3 + 1.5IQR \cdot e^{-3MC}]; & MC < 0 \end{cases} \tag{4.6}$$

where Q_1 is the 25th quantile, Q_3 is the 75th quantile, $IQR = Q_3 - Q_1$ and MC

Table 4.5: Table showing the distributions that best fit data

Comp. level c	Distribution $f_c(s)$	Notes
"02-03"	Normal($\mu = 0$, $\sigma = 0.06557143$)	Prob. mass is 50%
"04-05"	Normal($\mu = 0.09644720$, $\sigma = 0.05115680$)	Prob. mass is 97%
"06-07-08"	Normal($\mu = 0.20037248$, $\sigma = 0.05115680$)	
"09-10"	Normal($\mu = 0.25586938$, $\sigma = 0.05568770$)	
"11-12"	Normal($\mu = 0.32078588$, $\sigma = 0.07019838$)	

is the *medcouple* of data belonging to level c. Scores not falling in these intervals are discarded as outliers. After removing the outliers, we can now study the score distributions.

Let's define $f_c(x)$ as the density function (DF) of a probability distribution approximating the distribution of the score s for the complexity level c, in the "cleaned" corpus. To find $f_c(s)$, we used the Kolmogorov-Smirnov test that pointed out the most promising functions. Then, we selected the best fitting function by comparing the behavior of a candidate $f_c(s)$ and its Cumulative Density Function (CDF), against the DF and CDF reconstructed from the corpus data.

Figure 4.7 illustrates the results of this procedure, where DF of the corpus data is represented as a histogram, while its CDF is drawn as a curve; notice that the CDF reconstructed from the corpus data and the candidate distribution CDF are nearly indistinguishable. Table 4.5 displays the parameters of the identified $f_c(s)$.

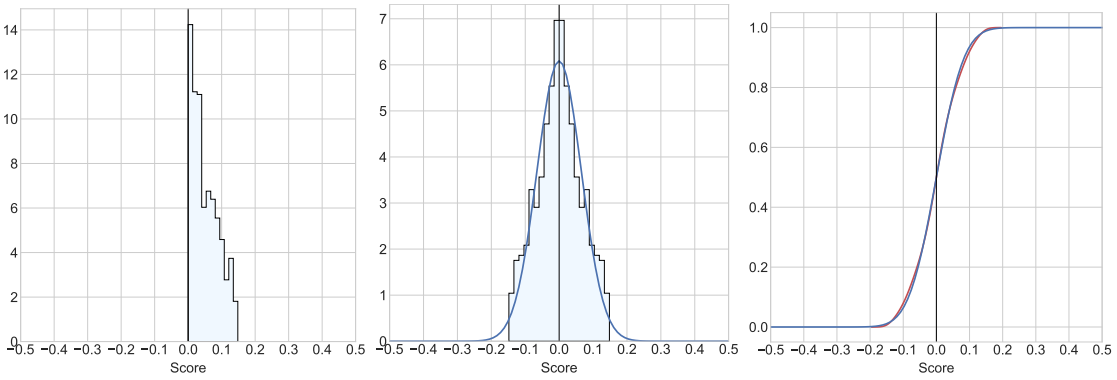
For level "02-03", we realized that data actually depicted half of a Normal distribution centered in zero (see Figure 4.7a). So, we "mirrored" data around zero and found, as expected, that the best fitting distribution was a Normal. Notice, however, that, considering only $s > 0$, the DF area only accounts for 50% of the probability mass. This affects how Equation 4.7 will be defined (i.e., we double the probability mass associated to "02-03").

For class "04-05", the Normal DF accounts for 97% of the probability mass (as part of the mass is distributed to $s < 0$). This is a small error, however, and we chose to ignore it.

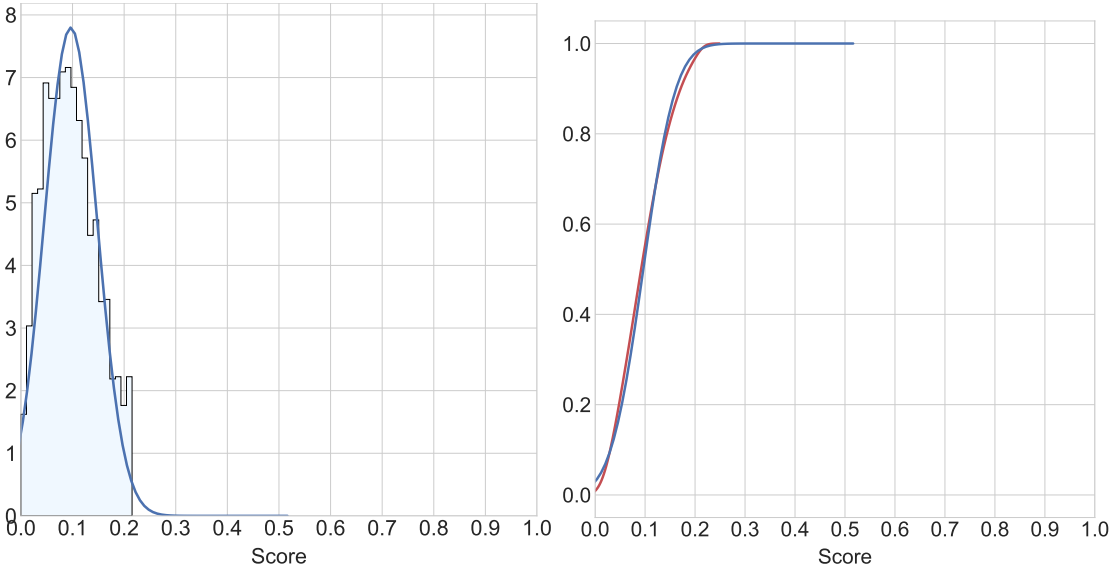
Then, being $f_c(s)$ a continuous distribution, the likelihood that an interval centered in s, with radius δ, belongs to the complexity level c is

$$\begin{gathered} L(s-\delta < x < s+\delta | c) = \left(1 + \mathbb{1}_{\{\text{"02-03"}\}}(c)\right) \cdot \int_{s-\delta}^{s+\delta} f_c(x)dx \\ \forall c \in S_c = \{\text{"02-03"}, \text{"04-05"}, \text{"06-07-08"}, \text{"09-10"}, \text{"11-12"}\} \end{gathered} \tag{4.7}$$

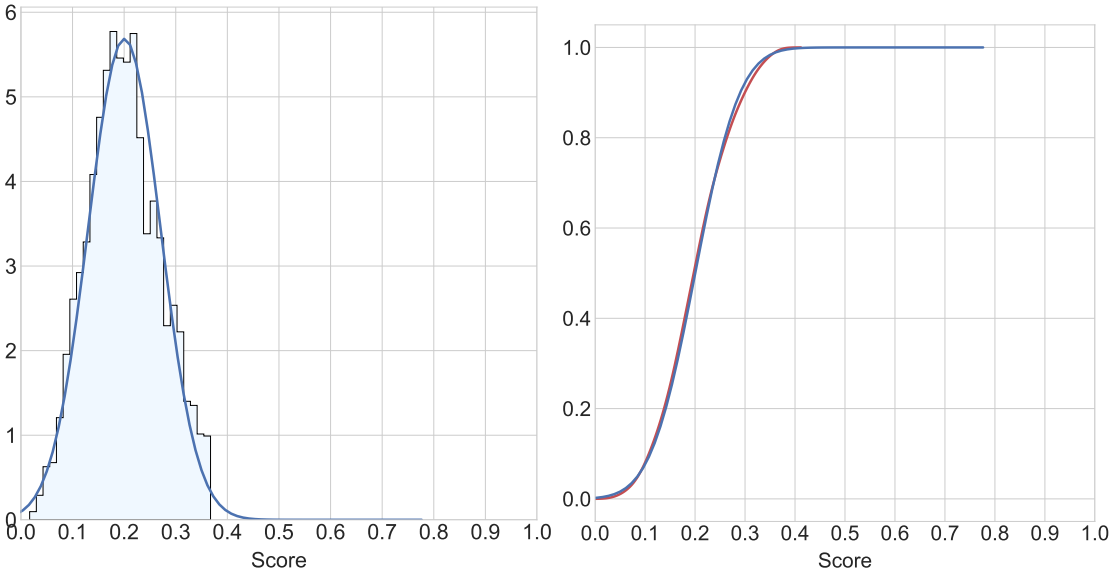
where S_c is a totally ordered set (and the order relation is obviously defined) and $\mathbb{1}_{\{\text{"02-03"}\}}(c)$ is the indicator function. We can then define $K(s)$, a function that



(a) True histogram, DF and CDF of level "02-03"

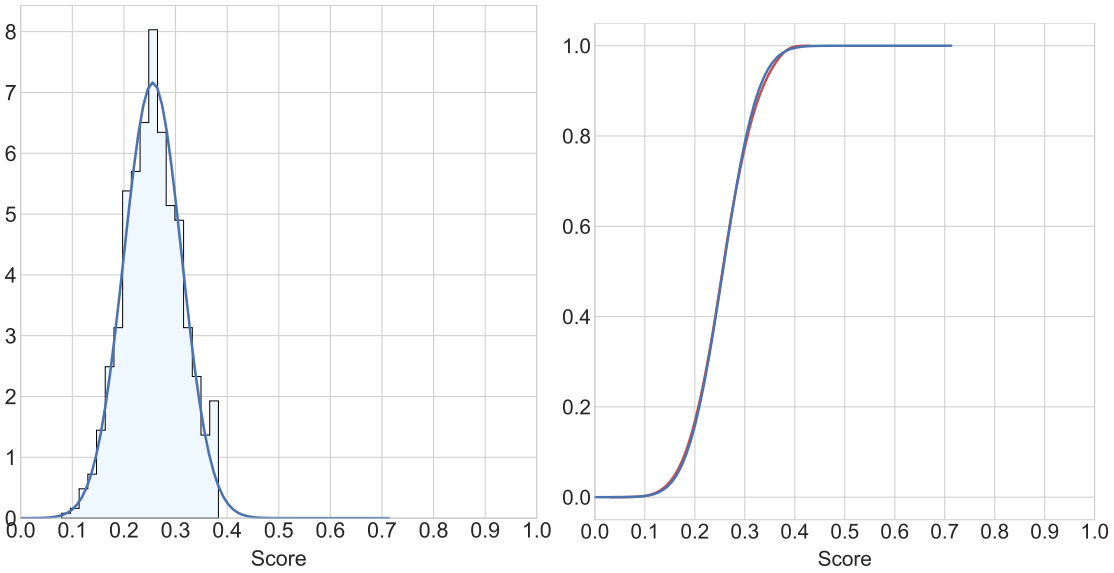


(b) DF and CDF of level "04-05"

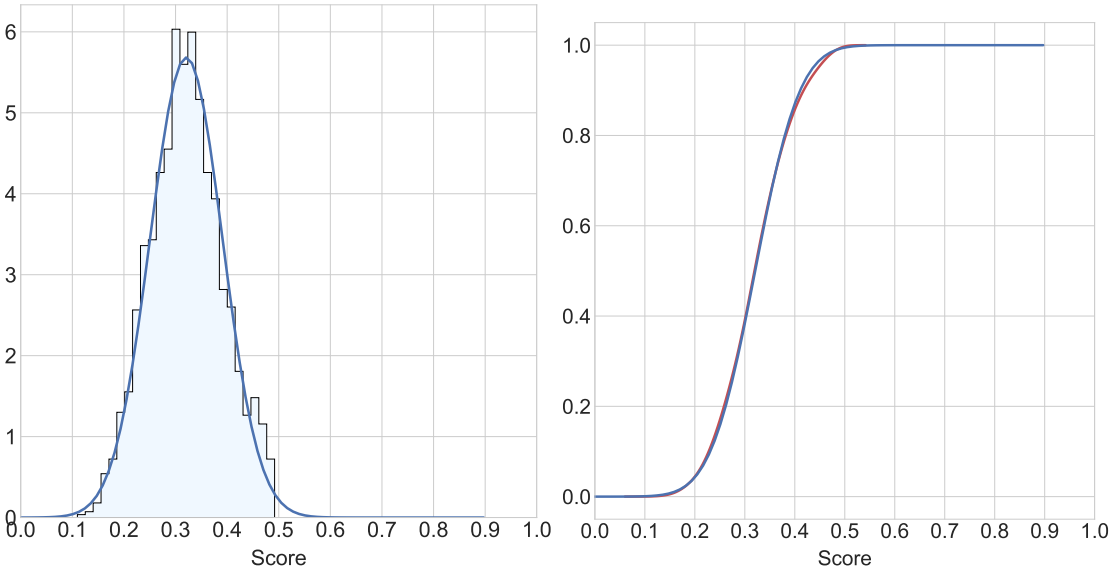


(c) DF and CDF of level "06-07-08"

Figure 4.7: For every complexity level (Newsela corpus): histogram of the data DF; distribution $f_c(s)$; CDF for both data and distribution.



(d) DF and CDF of level "09-10"



(e) DF and CDF of level "11-12"

Figure 4.7: For every complexity level (Newsela corpus): histogram of the data DF; distribution $f_c(s)$; CDF for both data and distribution.

associates a given score s to a complexity level, $K(s) : [0, 1] \subset \mathbb{R} \to S_c$; then, the probability of s belonging to a complexity level c is

$$P_\delta(K(s) = c) = \frac{L(s - \delta < x < s + \delta | c)}{\sum_{c' \in S_c} L(s - \delta < x < s + \delta | c')} \tag{4.8}$$

and, if authors are required to produce texts with complexity level no higher than $\tilde{c}$, the probability that the complexity level associate to a given score is not higher than $\tilde{c}$, is

$$P_\delta(K(s) \leq \tilde{c}) = \sum_{c' \leq \tilde{c} \in S_c} P_\delta(k(s) = c'). \tag{4.9}$$

Figure 4.8, on top, shows graphs of $P_\delta(K(s) = c)$; then, we can define our scale calculating the most likely complexity level of score s, as

$$\hat{c} = \arg\max_{c \in S_c} P(K(s) = c) = \arg\max_{c \in S_c} L(s - \delta < x < s + \delta | c). \tag{4.10}$$

Given the thresholds $[s_0 = 0, \ldots, s_5 = 1]$ for s, we can calculate s' taking values in a scale with equidistant thresholds $[s'_0 = 0, s'_1 = 0.2, s'_2 = 0.4, s'_3 = 0.6, s'_4 = 0.8, s'_5 = 1]$, maybe more convenient (although the complexity "density" changes in each level of the scale). Given s and s', and being (s_{i-1}, s_i), (s'_{i-1}, s'_i) the borders of the complexity level they belong to (of course, s and s' belong to the same complexity level), the relationship between the two scales is

$$\frac{s_i - s_{i-1}}{s'_i - s'_{i-1}} = \frac{s - s_{i-1}}{s' - s'_{i-1}}. \tag{4.11}$$

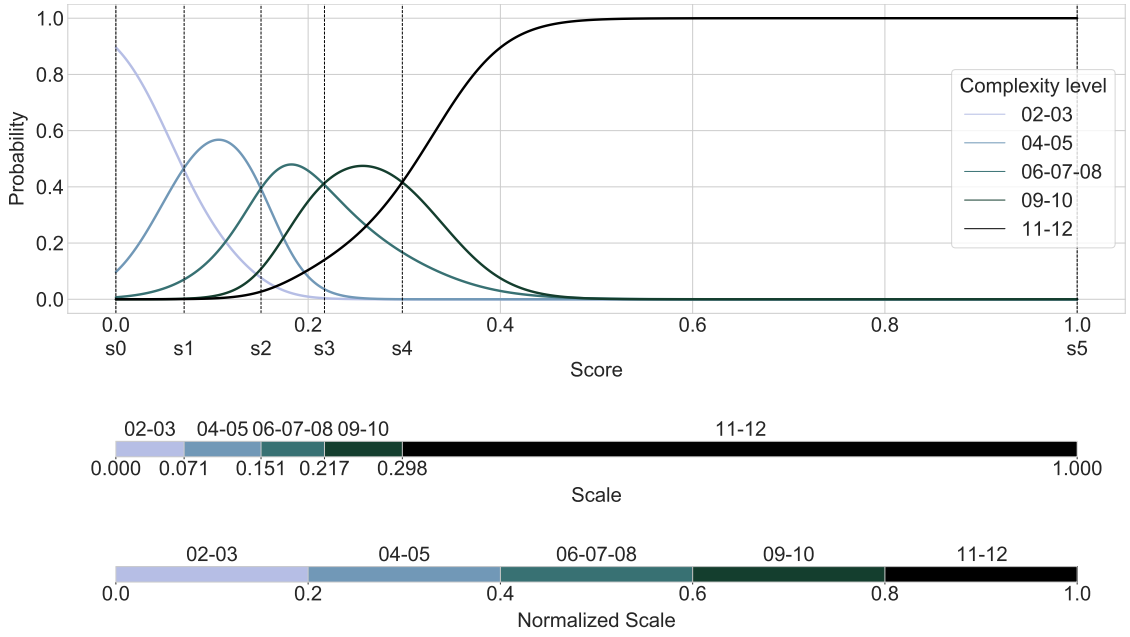


Figure 4.8: Probability that the score s belongs to a complexity level ($\delta = 10^{-6}$); scales for s and s'

A visual representation of the two scales, for s and s', is provided on the lower part of Figure 4.8.

A precision on the current approach has to be done, as mentioned Equation 4.7, it is the results of an approximation made on the behavior of distribution "04-05", however, if such approximation is considered unacceptable then the equation can be proposed in a generalized fashion as:

$$L(s-\delta<x<s+\delta|c)=\left(1+\sum_{c'}\frac{e_{c'}}{1-e_{c'}}\mathbb{1}_{\{c'\}}(c)\right)\cdot\int_{s-\delta}^{s+\delta}f_c(x)dx \tag{4.12}$$

where, for each complexity level c', $e_{c'}$ is the fraction of the $f_{c'}(s)$ probability mass that does not belong to the $[0,1]\subset\mathbb{R}$ range.

It is easy to show that Equation 4.12 is still a CDF in $[0,1]\subset\mathbb{R}$:

$$\left(1+\sum_{c'}\frac{e_{c'}}{1-e_{c'}}\mathbb{1}_{\{c'\}}(c)\right)\cdot\int_{0}^{1}f_c(x)dx=1 \tag{4.13}$$

We can solve the equation for every c'

$$\begin{aligned}\left(1+\sum_{c'}\frac{e_{c'}}{1-e_{c'}}\mathbb{1}_{\{c'\}}(c)\right)\cdot\int_{0}^{1}f_c(x)dx=&\\ &=\left(1+\frac{e_{c'}}{1-e_{c'}}\right)\cdot(1-e_{c'})\\ &=1-e_{c'}+e_{c'}\\ &=1\end{aligned}$$

In particular, considering $c'\in\{$"02-03", "04-05"$\}$ and $e_{c'}\in\{0.5,0.03\}$ we can improve the approximation provided by Equation 4.7 and thus the probability provided by Equation 4.8.

Chapter 5

Syntactic Approach

In this section we will present our approach to compute Syntactic Complexity, firstly, we will identify the features that better approximate the complexity by mean of classification task performance analysis. Secondly, the best set of features is selected, and used in a regression mechanism to obtain the Syntactic Complexity for every text sample. Lastly, the obtained score will be analyzed and the best performing model will be selected.

The steps will be implemented using two different models: Multi-Attentive model and Multi-Hierarchical model. These models were selected because their nature is prone to focus more on the syntactic aspects of a document. Both models will be presented in the next two sections.

5.1 Multi-Attentive model

The Multi-Attentive model was first introduced in [5], where it was presented as an automatic multilingual readability assessment technique.

This model, like many others introduced for the task, considers words as its base input; however, internally, the network is supplied with additional information, such as Morphology and POS tag, that will help it performs a better choice when applying the attention mechanism. The innovation introduced by this model is the attention mechanism itself; for the first time, a Multi-Attentive model is proposed, allowing the network to better discern and focus on specific features of the text sample while executing its learning task. (See Section 2.5.2 for more details)

The model has been re-adapted to the current objective by completely removing the lexical information as input, increasing the diversification of tested inputs, its modularity, and generally improving the performance of the network.

From here on, when referring to the model, we will describe our variant of the proposed mechanism.

Table 5.1: Example of relationship between sentences and sequences

Sentence	Mary	likes	dogs	and	cats
Sequence (UPOS)	PROPN	VERB	NOUN	CCONJ	NOUN

5.1.1 High Level Architecture Overview

The model can be ideally divided in four layers: Input, RNN, Attention, and Output.

Input: As the name suggests, this layer is responsible for handling and parsing the input that will then be fed to the network. Every input will be parsed and translated into its corresponding dense representation, also defined as Embedding (see Section 2.4.1)

We provided the network with up to 4 possible types of input (Morphology, UPOS, XPOS, and Dependency Path), all explained in Section 2.2.2. We will refer to the representation of this type of input as *Element*, a series of elements will define a *Sequence*. This definition is analogous to the concept of *Word* and *Sentence*; while a word is the base unit of a sentence, an element is the base unit of a sequence. To better understand the relationship between words - sentences, and elements - sequence we can look at Table 5.1

While generating the embedding for UPOS and XPOS, is standard practice, since it is a simple translation to a different representation system, it is not so common to apply the embedding also at the morphological and dependency level. These kinds of data are represented by a list of attributes; for this reason, they are converted beforehand in a single string and then translated to the Embedding representation.

RNN: Recurrent Neural Networks constitute the second layer, and they are responsible for creating a representation of the inputs that will be used later on by part of the attention mechanism. In particular, as discussed in Section 2.4.2, to overcome the problem of the vanishing gradients, Bidirectional LSTMs (BiLSTM) are used.

Every input is parsed independently, this means that the BiLSTMs do not share information, allowing the networks to focus only on a specific type of data.

Attention: Attention Layer is the most important component of the entire mode. As widely described in Section 2.4.3, attention is a mechanism used to simulate the ability of humans to focus on specific information.

In this model, the attention mechanism works both at the sequence and element level or, in other terms, it can identify both the most relevant elements and sequences in a document.

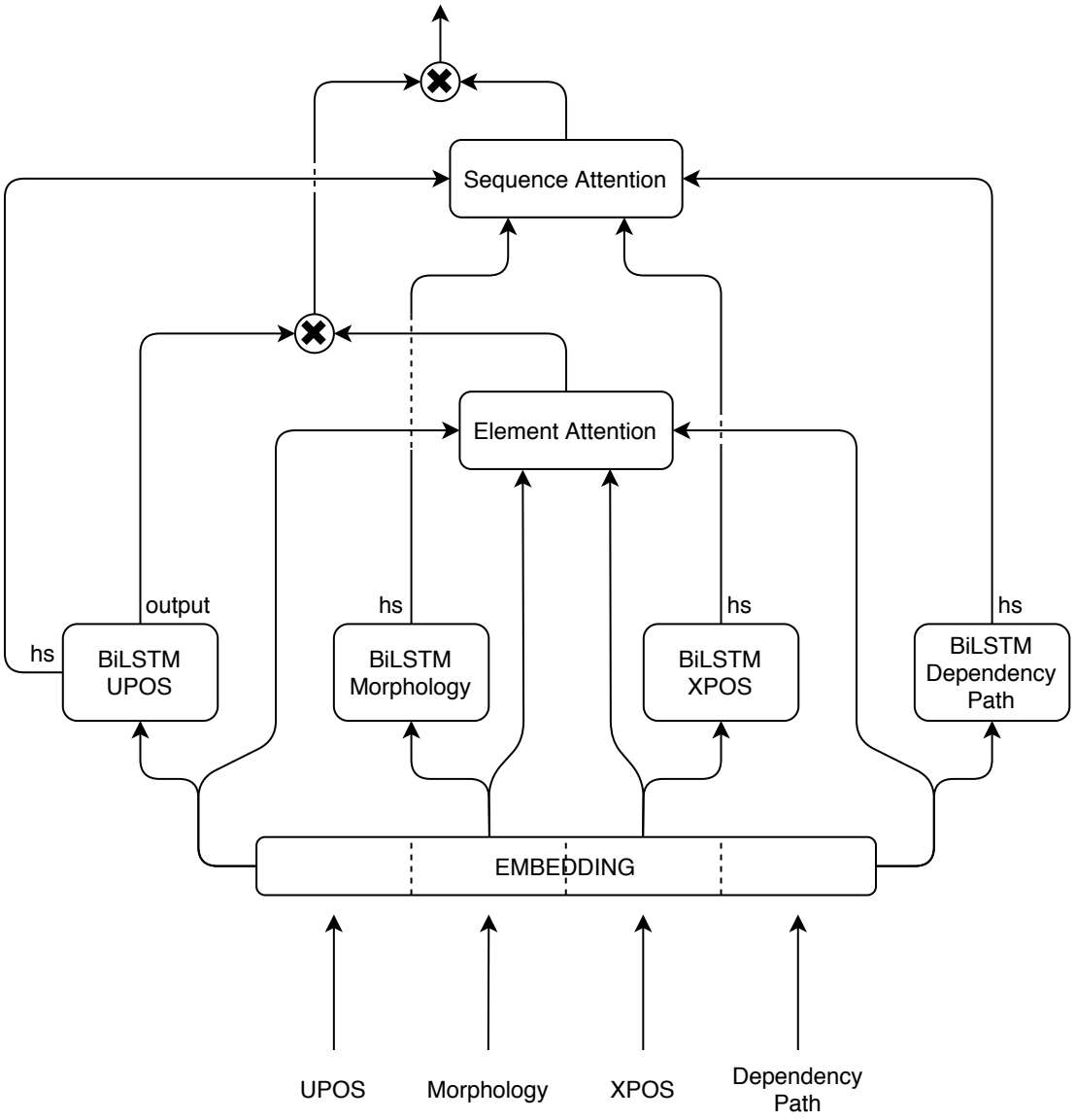


Figure 5.1: General architecture of the Multi-Attentive model

Continuing the diversification introduced in the RNN layer, also in the attention layer, multiple attention systems exist, each handling a specific kind of input.

Output: This is the last layer and is the result of the application of element and sequence attention on the output of the UPOS BiLSTM. The final output will either be a series of the probability of belonging to a certain class (if a discrete prediction is done), or value (if the task requires a continuous prediction).

Figure 5.1 displays a high level representation of the model.

5.1.2 Deep Level Architecture Overview

Here, we are going to present all the four layers mentioned in the previous section, in deeper detail, by also providing mathematical representation of the operations used.

Let's define a document d as the input for the model, d is in reality composed by the four varieties of input introduced into the network: $d =< d_u, d_x, d_m, d_d >$, where u, x, m, d stands for UPOS, XPOS, morphology, and dependency path. For the sake of simplicity from now on we are going to refer to a general d_T without specifying, unless relevant, the subtype considered. It is, however, important to remember that the model parses every input independently.

For Every input we can reach the sequence level by using $d_{T,i}$, identifying the i_{th} sequence in document d, and $d_{T,ij}$ when referring to the j_{th} element of the i_{th} sequence in document d.

The first step in parsing the input d_T, consists in translating the document from the standard representation to the dense version, this translation can be imagined as the equivalent of a look up table $\Omega_T \in \mathbb{R}^{v \times d}$ in which each row is an embedding for a specific element in the vocabulary of size v, represented by d features. Needless to say, every variety of input is associated to a different look up table, for this reason v and d might differ among subtypes. We can define the resulting dense representation as ω_T.

After pre-processing the input data, the representation of the document is feed to the RNN model. The output of the BiLSTM, obtained by concatenating the final states of the network, can be defined as $h_{T,i}$ if we are considering a sequence, or as $h_{T,ij}$ if we are considering a specific element.

Parallel to this activity, it is also computed the attention, as mentioned before both at element and sequence level.

The element-level attention is composed of multiple attention mechanisms aggregated, with each block following the same structure, differing only for the size of the input and the number of hidden units. Each attention block is a two-layers fully connected neural network and can be represented by the following formulas:

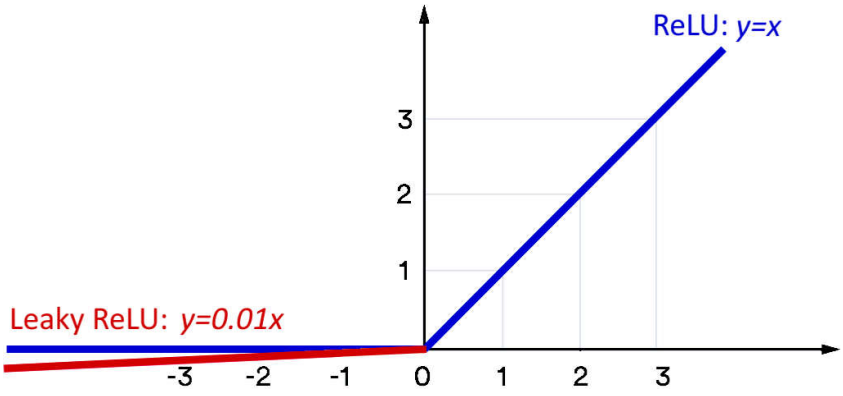


Figure 5.2: ReLU and Leaky ReLU activation functions

$$att_{T1,ij} = \text{ReLU}(w_1 \cdot \omega_{T,ij} + b_1) \tag{5.1}$$

$$att_{T2.ij} = \text{ReLU}(w_2 \cdot att_{T1,ij} + b_2) \tag{5.2}$$

where w and b represents the weights and biases of a linear layer, $\omega_{T,ij}$ is the dense vector representation of the j_{th} element in the i_{th} sequence of the input to the network, and "ReLU" represents the Rectified Linear Unit activation function. Mathematically, the ReLU is equivalent to $y = max(x, 0)$, and a visual representation is available in Figure 5.2.

Lastly, after applying a mask to the attention, in order to ignore the padding introduced by the RNN, every attention is multiplied by a weighted value $z_{T,norm}$, automatically estimated during the training phase, such that $\sum_{t \in T} z_{t,norm} = 1$. This condition is forced by applying a softmax to the value of z:

$$z_{T,norm} = \frac{exp(z_T)}{\sum_{t \in T} exp(z_t)} \tag{5.3}$$

The final attention a_{ij}, associated to the element d_{ij} is then computed as:

$$a_{ij} = \sum_{t \in T} z_{t,norm} \cdot att_{t2,ij} \tag{5.4}$$

In a similar way it is possible to generate the sequence level attention (a_i), by simply using the hidden representation retrieved by the BiLSTM $(h_{T,i})$ instead of the dense representation $\omega_{T,ij}$. Figure 5.3 displays a schematic representation of the attention mechanism.

In conclusion, the two attentions are multiplied to the output of the BiLSTM responsible for the UPOS representation, and a final h_{out} is obtained. This output will be mapped in the output layer either by applying a softmax activation function, if it is required a discrete prediction or, by implementing a "Leaky ReLU" activation

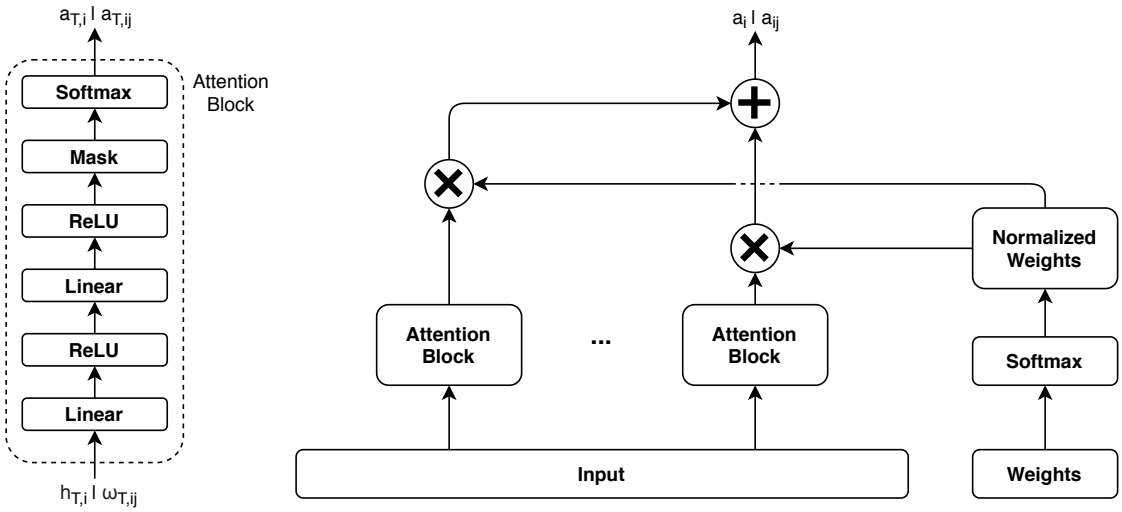


Figure 5.3: Schematic representation of the attention mechanism for the Multi-Attentive model

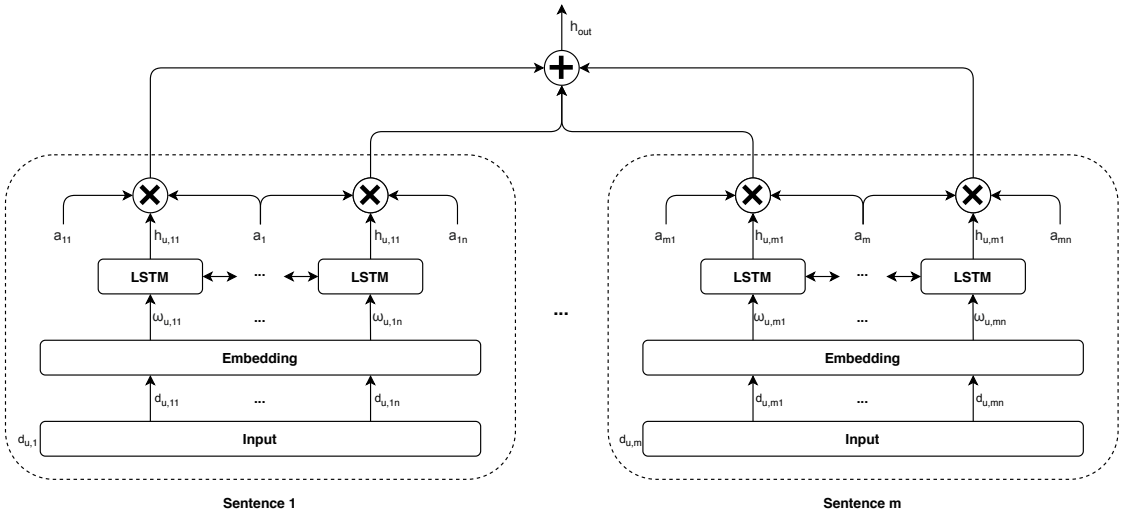


Figure 5.4: Detail schematic representation of the Multi-Attentive model

function if the objective is to execute a continuous prediction. "Leaky ReLU" is a variant of ReLU, with a small slope for negative values, instead of altogether zero (See Figure 5.2).

Figure 5.4 presents the detailed structure of the model

5.2 Multi-Hierarchical model

The Hierarchical Attention Network (HAN) was first presented in [6], in which it was introduced as an innovative approach. The researchers, following the idea that "words form sentences and sentences form documents", built a network model

that resembles the structure of the documents. This is achieved by generating first a sentence representation, and then, leveraging such representation to generate the document representation.

Also in this model, the implementation of an attention mechanism is at two different levels aiming to handle the unequal importance of sentences and words in a different part of the document.

Figure 5.5 displays the original architecture. It can be ideally divided into four layers, responsible for encoding and attention operations for both word and sentence level. Both word encoder and sentence encoder are based on GRU, introduced in [2], that can comprehend the states of sentences without using additional memory cells (see Section 2.4.2).

The only difference between the two encoders is the type of input, in one case being the dense representation of the text sample and the other the results of the application of the word attention level. The attention mechanisms instead are based on the analysis of the similarity between the representation of the word and a vector representing the context (defined as u_w and u_s, respectively for word level and sentence level attention).

The model was initially presented for sentiment estimation and topic classification tasks. We re-adapted it to the current job and converted to integrate also the innovation proposed by the Multi-Attentive model. From here on, when referring to the model, we will describe our implementation of the proposed mechanism.

5.2.1 High Level Architecture Overview

The model can be ideally divided into six layers: Input, Element Encoding, Element Attention, Sequence Encoding, Sequence Attention, and Output.

Input: This layer, similarly to the one presented for the previous model, is responsible for handling and parsing the input that will be fed to the network. Every input will be translated to the respective dense representation (Embedding, see Section 2.4.1.)

As in the previous model, we provide up to four possible varieties of input (Morphology, UPOS, XPOS, and Dependency Path), see Section 2.2.2 for details. These inputs will be processed in an analogous way to the Multi-Attentive model. We will also maintain the same notation presented before, using *Element* and *Sequence*, to describe the network representation of *Word* and *Sentence*.

Element Encoding: Element Encoding represents the second layer and is responsible for generating a representation of the input. Such representation will be used by the next layer to compute the attention at the element level. This layer is usually composed of RNN networks and, as a consequence of the discussion in Section 2.4.2,

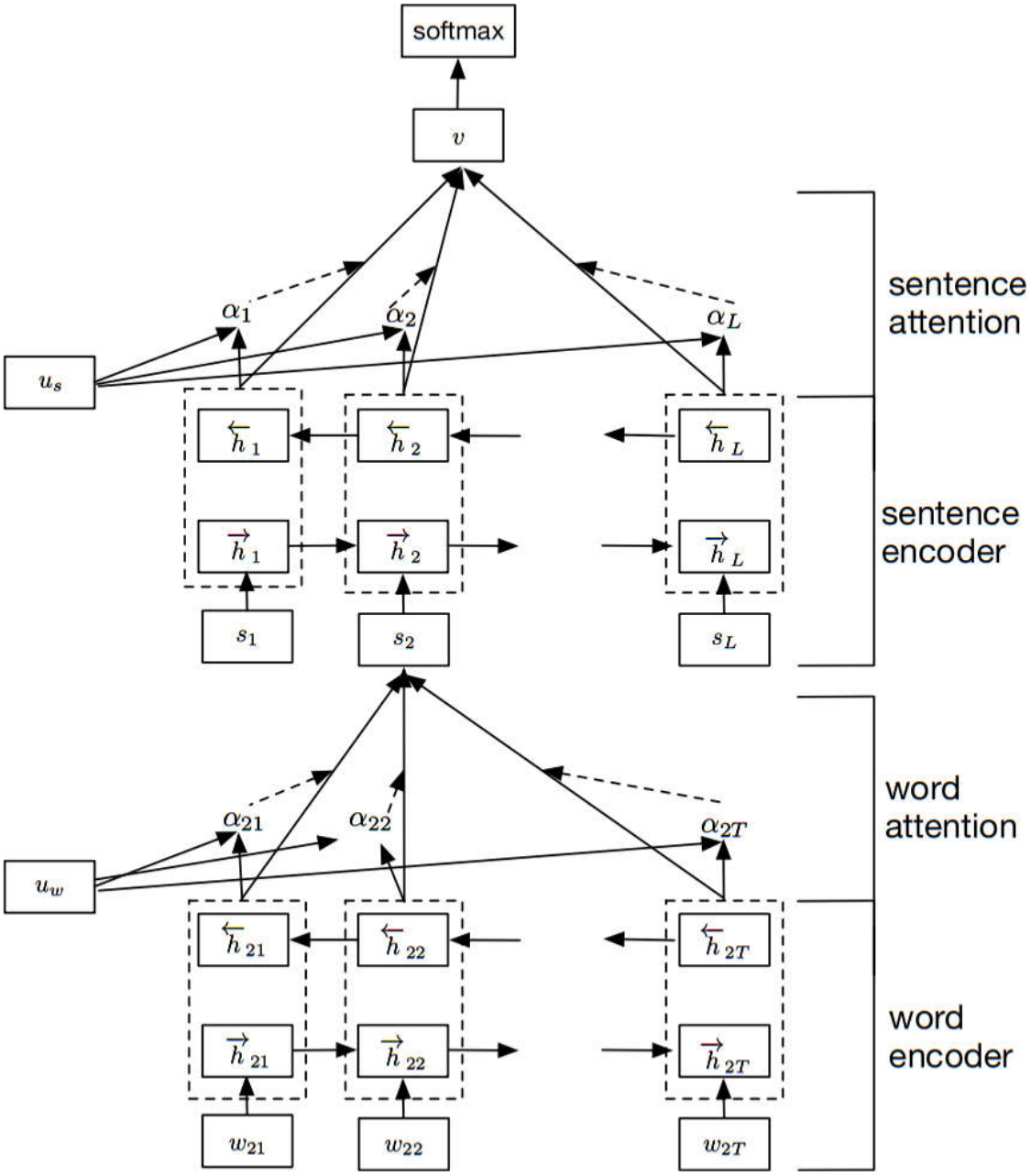


Figure 5.5: Hierarchical Attention Network as proposed in [6]

to overcome the problem associated with the vanishing gradient, BiLSTMs are used.

Again, every type of input is parsed independently, this means that the BiLSTMs do not interact with each other, granting a diversification of the learning process.

Element Attention: Element Attention layer consists of the attention mechanism applied at the element level, the input to this step is the output of the previous layer and, as mentioned in Section 2.4.3, tries to emulate the innate capacity of humans to focus only on part of the information while solving a problem.

Following the approach started in the previous layers, also, in this case, the attention mechanisms for different input types are completely independent.

Sequence Encoding: Sequence Encoding layer presents the same structure as the Element Encoding layer, with the only difference being the format of the input. Instead of using the dense representation of the input to the network as input for the layer, it uses the result of the application of the Element Attention layer over the Element Encoding layer.

Since the structure is the same, also in this layer persists the diversification among the various types of input.

Sequence Attention: Sequence Attention, as can be imagined, follows the same structure of the Element Attention layer, with the only difference being, also in this case, the type of input to the layer. Sequence Attention uses as input the output of the Sequence Encoding layer.

This is the last layer in which the diversification among the input is maintained. In the next layer, the different processes will be merged to provide a single output.

Output: This is the last layer of the model, it is also the layer in which the diversification in handling the inputs is terminated. After applying to every Sequence Encoding the computed Sequence Attention, the results will be merged providing a single final result. The final output can be either a series of the probability of belonging to a certain class if the intended task is a classification (discrete prediction), or a single value if a regression is the objective of the model (continuous prediction).

Figure 5.6 displays a visual high level representation of the model.

5.2.2 Deep Level Architecture Overview

In this section, we are going to cover all the previously mentioned layers, by presenting each one in deeper details supported by a mathematical representation of the steps.

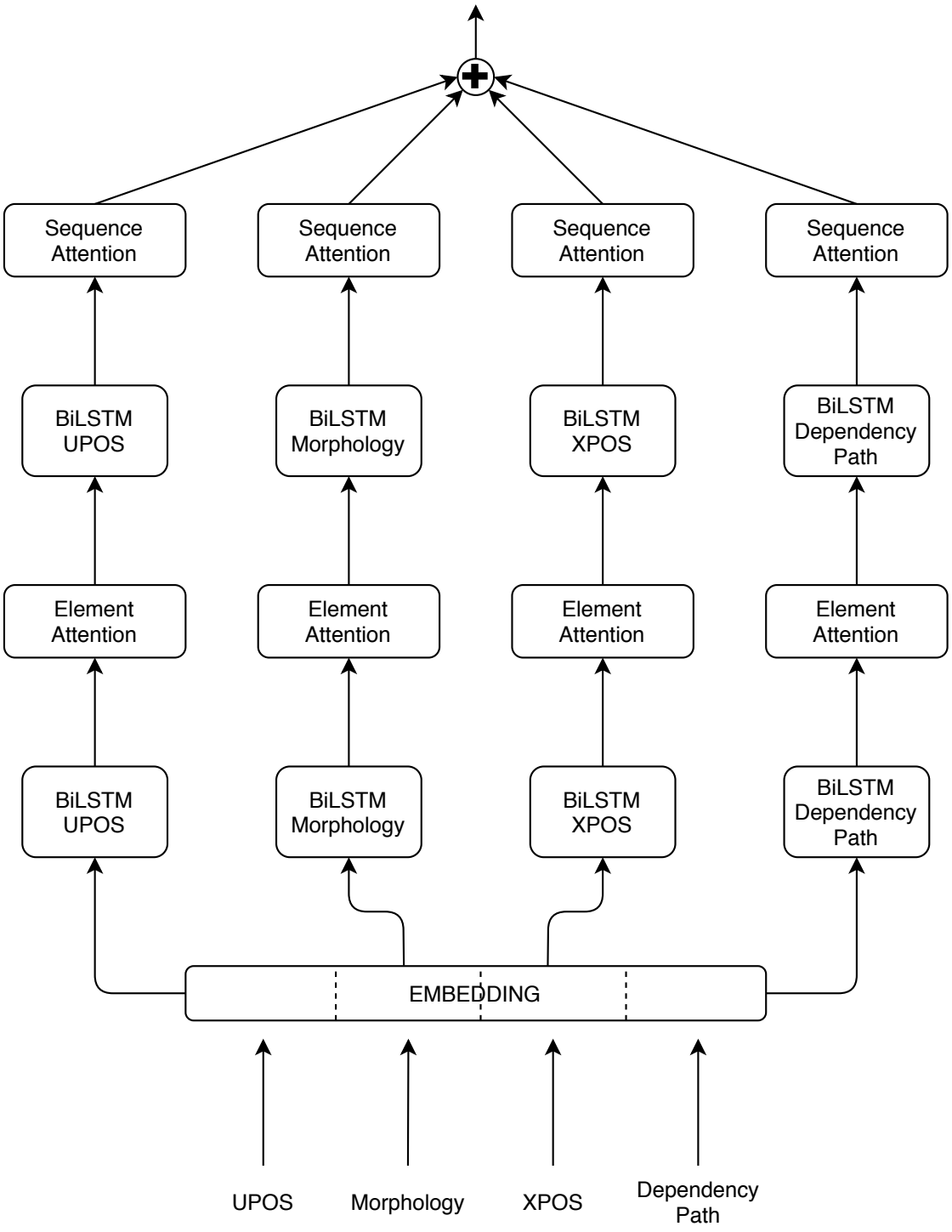


Figure 5.6: General architecture of the Multi-Hierarchical model

The first layer, is the same as the Multi-Attentive model, for this reason, we can maintain the same definition and conclusions, here reported for convenience.

Let's define a document d as the input for the model, d is in reality composed by the four varieties of input introduced into the network: $d =< d_u, d_x, d_m, d_d >$, where u, x, m, d stands for UPOS, XPOS, morphology, and dependency path. For the sake of simplicity, from now on we are going to refer to a general d_T without specifying, unless relevant, the subtypes considered. It is, however, important to remember that every input is parsed independently.

For Every input we can reach the sequence level by using $d_{T,i}$, identifying the i_{th} sequence in document d and $d_{T,ij}$ when referring to the j_{th} word of the i_{th} sequence in document d.

The first step in parsing the input d_T, consists in translating the document, from the standard representation to the dense version. This translation, can be imagined as the equivalent of a look up table $\Omega_T \in \mathbb{R}^{v \times d}$, in which each row is an embedding for a specific word in the vocabulary of size v, represented by d features. Needless to say, every variety of input is associated to a different look up table, for this reason v and d might differ. We can define the resulting dense representation as ω_T.

After pre-processing the input, the dense representation is fed to the element encoding layer. The output of the BiLSTM (defined as $h_{T,ij}$) is then given as input to the element attention layer.

In a similar fashion to the previous model, the element attention layer is composed of multiple independent and almost identical structures. Every block can be imagined as an independent network system, and, as such, can be simply expressed in the mathematical formula as:

$$att_{Tw1,ij} = \tanh(w_1 \cdot h_{T,ij} + b_1) \tag{5.5}$$

$$matt_{Tw1,ij} = \text{mask}(att_{Tw1,ij}) \tag{5.6}$$

$$att_{Tw2,ij} = \sigma(w_2 \cdot matt_{Tw1,ij} + b_2) \tag{5.7}$$

where w and b are the weights and biases of a linear layer, $h_{T,ij}$ is the output of the BiLSTM of the previous level and it is the input to the attention. "tanh" represent the hyperbolic tangent activation function, "mask" perform a simple masking of the data removing references to the padding added by the element encoding layer, and σ indicates a softmax normalization applied to the output of the attention block before returning the result.

The hyperbolic tangent (tanh) is a continuous function that produces an output between -1 and 1 for every given x as input, mathematically it is defined as:

$$tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \tag{5.8}$$

Softmax, instead, is a function that takes as input a vector z of K real numbers, and normalizes it into a probability distribution consisting of K probabilities proportional to the exponential of the input number. In other words, starting from a series of values that can be either positive and negative, it returns for each value, an equivalent representation between 0 and 1, with the constraint that all returned values must sum up to 1. Mathematically is defined as:

$$\sigma(x)_i = \frac{\exp(x_i)}{\sum_{j=1}^{K} \exp(x_j)} \text{ for } i \in [1, \ldots, K], x = (x_i, \ldots, x_K) \in R^K \tag{5.9}$$

After computing the element level attention ($att_{Tw2,ij}$), the result is multiplied by the representation provided by the element encoding layer ($ha_{Tw,ij}$) and summed to generate the input for the sequence encoding layer ($ha_{Tw,i}$).

The sequence encoding layer act in the same way as the element encoding layer, with the only difference being the form of the input, passing from the simple dense representation of the input data to $ha_{Tw,i}$. The output of the BiLSTM, defined as $h_{Ts,i}$ is then given to the sequence attention layer. Again the procedure is identical to the previously mentioned one for the element attention level, with the only difference being the input.

The output of the sequence attention layer ($att_{Ts2,i}$) is then multiplied by the output of the sequence encoding layer ($h_{Ts,i}$), and summed to obtain the result for the entire document (ha_{Ts}).

Lastly, after obtaining the result for every type of input T, such results are multiplied by a weighted value $z_{T,norm}$. This value is automatically estimated during the training phase, and it is defined as:

$$z_{T,norm} = \frac{\exp(z_T)}{\sum_{t \in T} \exp(z_t)} \tag{5.10}$$

This definition ensures that $\sum_{t \in T} z_t = 1$.

The final output for a document d, is then computed as:

$$h_{out} = \sum_{t \in T} z_{t,norm} \cdot ha_{Ts} \tag{5.11}$$

This output will be mapped in the output layer either by applying a softmax activation function, if the desire is to compute a discrete prediction, or by implementing a "Leaky ReLU" activation function, if the objective is to execute a continuos prediction.

Figure 5.7 presents the detail structure of the model.

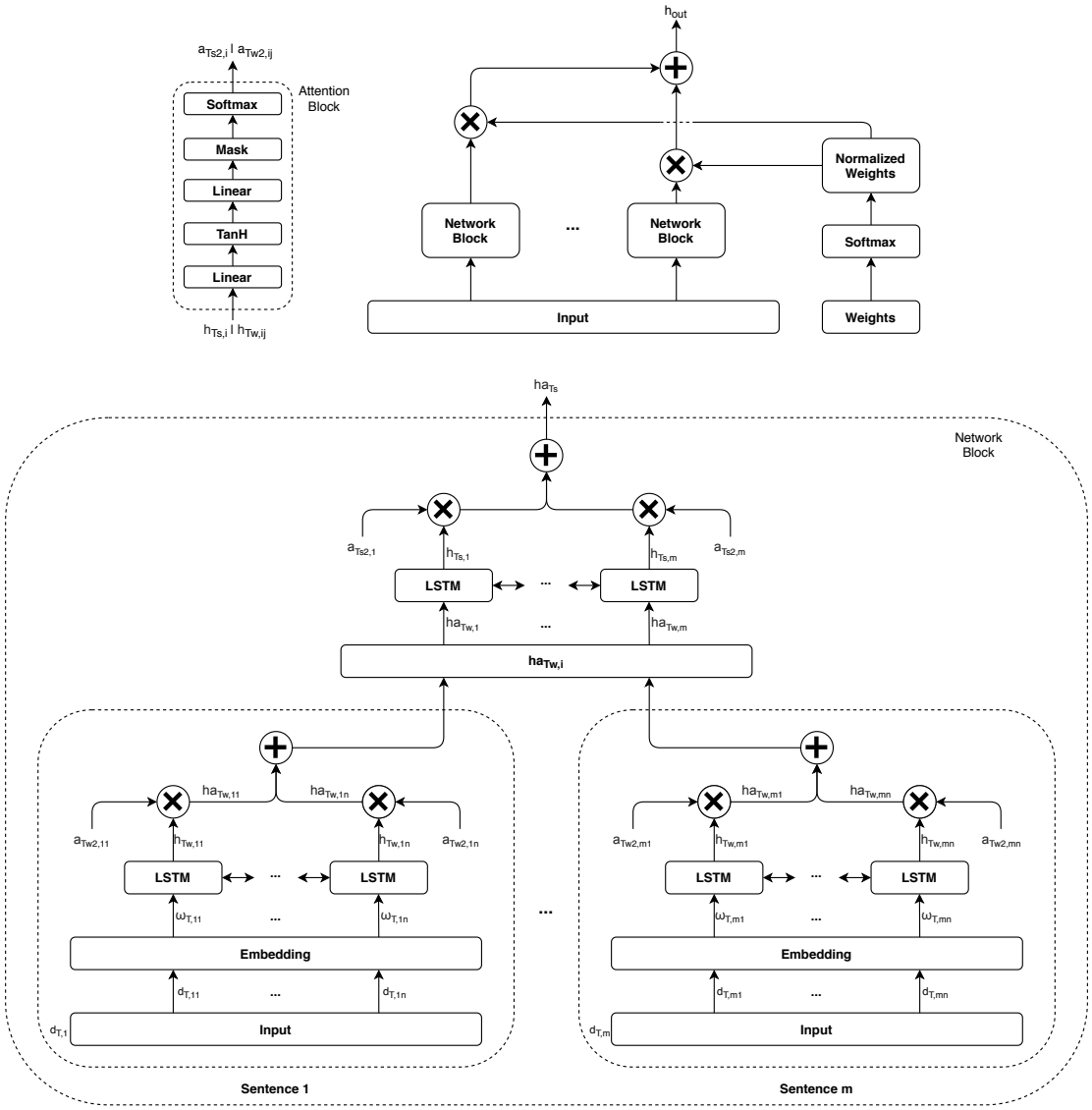


Figure 5.7: Detail schematic representation of the Multi-Hierarchical model

5.3 Experimental Results

In this section, we are going to explain how we implemented the models described in the previous sections and the obtained results. Following an approach similar to the lexical component, after analyzing the outcomes, we will verify the validity of the approach by using low-level indexes that capture multiple aspects of Syntactic Complexity.

5.3.1 Features Selection

The implementation of the two models is quite straightforward; however, before executing the two approaches, a preliminary step has to be taken: select the best set of features to use as input for each model.

To identify the highest performing input variety, we decided to implement the two models and perform a classification task. This task can give information about the type of features that better grant an effective diversification among the

Table 5.2: Results of classification tasks for Multi-Attentive model

Input type	Accuracy	Weighted Loss	Balanced Accuracy	Kappa
UPOS Morphology	0.6065	1.0041	0.5837	0.5430
UPOS XPOS	0.6178	0.9987	0.5920	0.5555
UPOS Dependency	0.6116	1.0253	0.5935	0.5491
UPOS Morphology XPOS	0.6229	1.0168	0.6027	0.5620
UPOS Morphology Dependency	0.6309	0.9977	0.6086	0.5711
UPOS XPOS Dependency	0.6084	1.0419	0.5811	0.5445
UPOS XPOS Morphology Dependency	0.6301	1.0160	0.6131	0.5706

Table 5.3: Results of classification tasks for Multi-Hierarchical model

Input type	Accuracy	Weighted Loss	Balanced Accuracy	Kappa
UPOS Morphology	0.7141	1.5969	0.6869	0.6671
UPOS XPOS	0.7033	1.6033	0.6748	0.6548
UPOS Dependency	0.6449	1.6374	0.6311	0.5880
UPOS Morphology XPOS	0.6309	1.6613	0.6063	0.5708
UPOS Morphology Dependency	0.6666	1.6213	0.6545	0.6131
UPOS XPOS Dependency	0.6931	1.6126	0.6639	0.6423
UPOS XPOS Morphology Dependency	0.6894	1.6029	0.6741	0.6394

complexity levels.

To further stress this aspect we used the original version of Newsela corpus, consisting of documents divided into 11 classes, from 02 to 12 of the Common Core Standard System. This dataset possesses the highest number of documents and the highest diversification on complexity levels, among the implemented corpora.

Tables 5.2 and 5.3 presents the results of this experiment, respectively for the Multi-Attentive model and the Multi-Hierarchical one.

The tables provide results considering: Accuracy (fraction of predictions our model got right), a widely used measure for the classification tasks, and three other metrics that take into consideration the uneven distribution of data among the classes. These three measures were introduced to diminish the interference produced by the strong diversity in the number of text samples that characterize every level of complexity. Such metrics are here briefly explained:

Weighted Loss : Weighted version of the classical loss metric (the prediction error of a network), this variant was introduced to tackle the problem of

imbalanced data, by assigning different weights when verifying the accuracy of a prediction for a certain class (the lower the better).

Balanced Accuracy : Balanced version of the classical accuracy, implemented as the average of the recall computed on each class, it has range between 0 and 1, where 1 is the best value and 0 is the worst.

Kappa : also called Cohen's kappa, expresses the level of agreement between two annotators on a classification problem, it is computed using the following formula $k = (p_o - p_e)/(1 - p_e)$, where p_o is the observed agreement ratio, and p_e is the expected agreement when both annotators assign labels randomly. [71, 72]

Both models are tested against all possible variants of input, with UPOS maintained for every implementation. While this condition is mandatory for the Multi-Attentive model, given its definition, it is not necessary for the Multi-Hierarchical model; however, to grant coherence between the tests we decided to maintain such constraint.

Considering both tables, we can notice that for the Multi-Attentive model, the best performing input type is the triplet: UPOS, Morphology, and Dependency. For the Multi-Hierarchical model, instead, the best performing input variety is the couple: UPOS and Morphology. These combinations will be used in the implementation of the model during the execution of the regression task.

5.3.2 Model implementation

Once defined the set of features that will be used with every model, the first step in implementing the algorithm is to select on which dataset train the two different models. This decision is very important because it will influence the quality of the returned result.

Ideally, this task must be driven by the need to find a corpus that has a high amount of syntactic diversification among the levels, or in other words, a corpus that has multiple documents of different genres. Given these premises, the obvious decision is to use Mixed corpus. It must be taken into consideration, however, that this corpus, is under the influence of the AppBCCS corpus, characterized by the huge presence of narrative documents, that might introduce graphical gimmicks which will negatively affect the model.

A second point that raises worries is associated with the strong different nature of documents in AppBCCS, compared to the other corpora that, instead, are mainly composed of articles. While this condition is considered optimal for the task at hand, it must be considered that text samples, belonging to AppBCCS, represent

only a small portion of the Mixed corpus. This disparity in the variety of text samples might generate noise in the form of a large number of outliers in the training process.

To ensure our model is not affected by the mentioned problems, we are going to implement both models using Mixed and Newsela corpus, generating four possible models: Multi-Attentive model trained on Mixed, Multi-Attentive model trained on Newsela, Multi-Hierarchical model trained on Mixed, and Multi-Hierarchical model trained on Newsela.

Each model, will then be tested against each of the five corpora previously presented. Figures 5.8 - 5.12 displays the result of this activity. In particular, every figure reports the results obtained by a specific corpus on the four different models listed above; for example, Figure 5.8 displays the results obtained by the Newsela corpus, while being used for validation on:

- The Multi-Attentive model trained on Mixed (Figure 5.8a)
- The Multi-Attentive model trained on Newsela (Figure 5.8b)
- The Multi-Hierarchical model trained on Mixed (Figure 5.8c)
- The Multi-Hierarchical model trained on Newsela (Figure 5.8d)

To avoid overlapping of data between the training and validation phase, when needed, we applied a 5-fold approach. The basic idea is to divide the dataset into five parts, selects four parts, merge them, and use them for training. The remaining share will be used for validation. This procedure is applied five times, selecting every time a different set of sections, until every section has been used for validation.

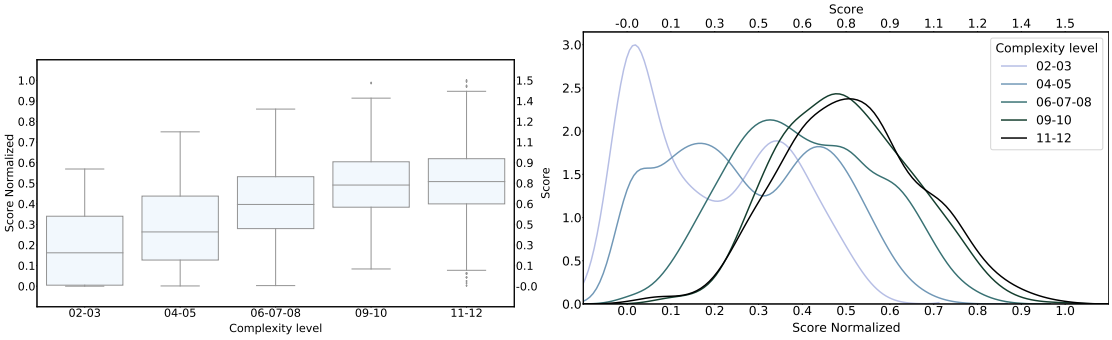
This approach is applied to Newsela, AppBCCS, and Mixed when the training is executed on Mixed. If the training is, instead, executed on Newsela, the approach is applied only for Newsela, and Mixed.

Similar to what has been done during the lexical section, every result is proposed in two forms, boxplot, and distributions, since from these two representations is possible to identify information concerning variance, density, and overlapping among the different complexity levels.

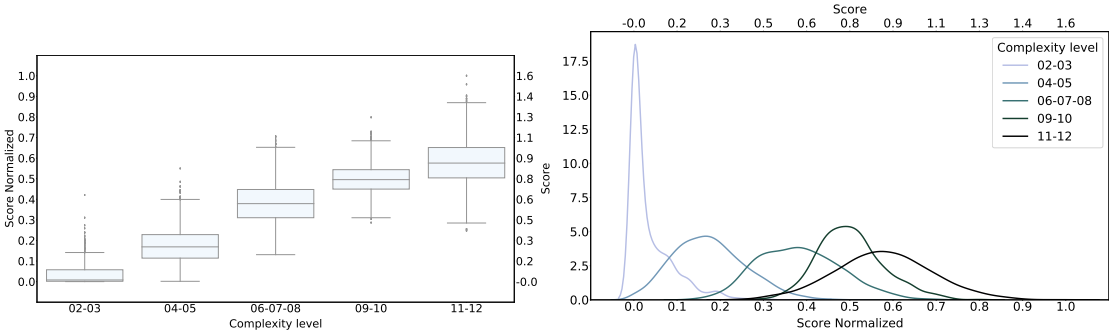
Lastly, it is relevant to notice that, given the nature of the networks models and the execution of the regression task, the returned values might be outside of the standard set [0,1], to avoid this, we applied a normalization of the retrieved score, with every corpus being normalized independently by the others.

Newsela

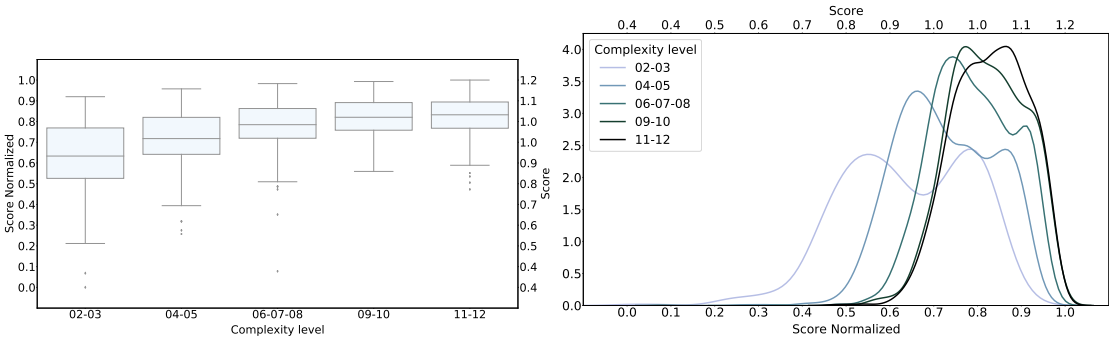
The first results presented, are generated by processing the Newsela corpus with both models and using both Mixed and Newsela corpora as a base for the



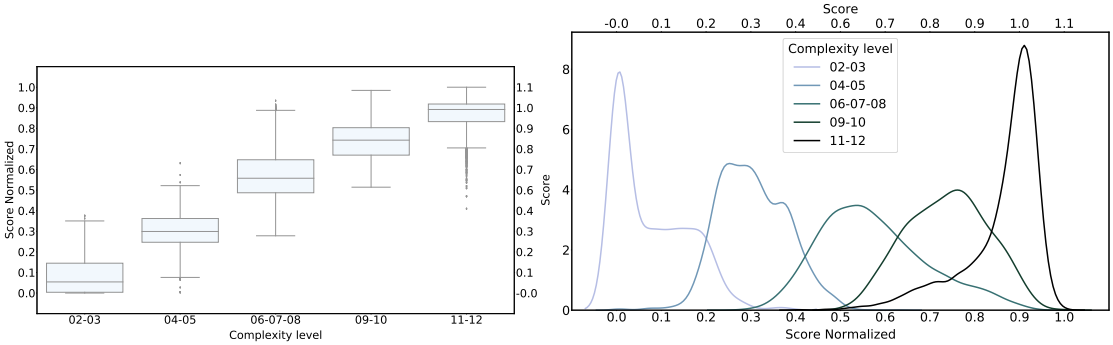
(a) Box plot and distribution using Multi-Attentive model with Mixed for training



(b) Box plot and distribution using Multi-Attentive model with Newsela for training



(c) Box plot and distribution using Multi-Hierarchical model with Mixed for training



(d) Box plot and distribution using Multi-Hierarchical model with Newsela for training

Figure 5.8: Box plot representation, and data distributions of the Newsela corpus

training; given the special nature of Mixed and Newsela corpora, in both cases, the results are retrieved applying a 5-fold approach, so to grant that there will never be overlapping between the training and validation dataset.

Independently by the model, it is evident the difference in the quality of the result between the approaches trained on Mixed and the ones based on Newsela, with the latter outperforming the former. The most probable source of this problem is the influence generated by the AppBCCS corpus in Mixed, confirming that the worries about the correctness of using Mixed as a training base might be grounded.

By going more in detail, in the approaches based on Mixed, the different complexity levels are subjected to a high level of overlap, in particular, the two highest ones. This phenomenon points out the inability of the model to properly discern among the most complex documents of the corpus. Furthermore, most of the levels present a double peak behavior, highlighting the difficulties of the model.

Contrary to the behavior showed in the approaches based on Mixed, the ones that use Newsela, display well distinct curves with no major anomalies, with the results provided by the Multi-Attentive model exhibiting a lower variance compared to the Multi-Hierarchical counterpart. The high variance is probably also the cause of the "steps" that appear in the lower levels of the Multi-Hierarchical approach.

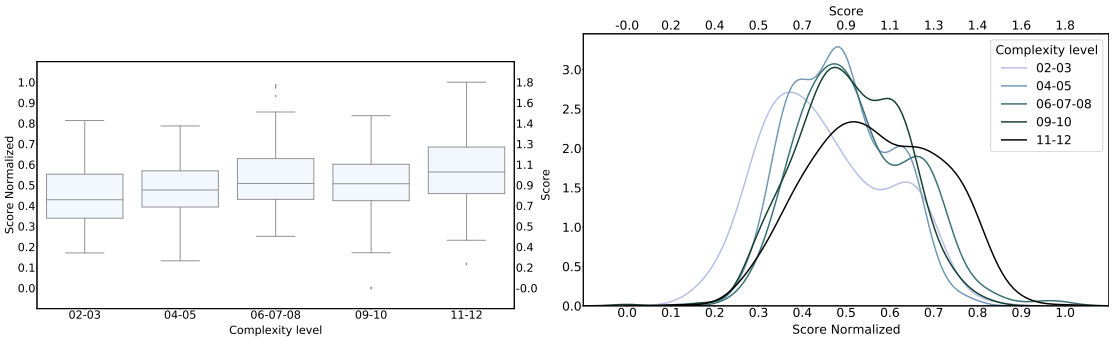
AppBCCS

The second datasets we tested is AppBCCS; in this case, the results are obtained using a 5-fold approach only when Mixed is used as a training base, otherwise, a standard approach is applied. Similar to what we noticed with Newsela, the models based on Mixed, generate worse results compared to the ones based on Newsela.

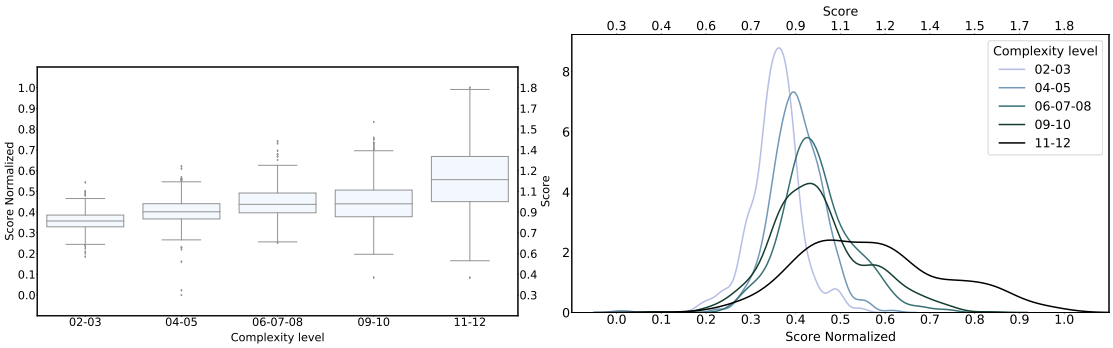
In particular, the models based on Mixed corpus, independently by the structure of the model, show a high degree of overlapping, with the curves being almost identical, and with the median almost stuck at the same value, independently by the complexity level.

A similar phenomenon also appears in the approaches based on Newsela, with the curves showing a high level of overlapping compared to the results obtained on Newsela (validation), however, in these cases, the medians of different complexity levels show a growing trend at the increasing of levels.

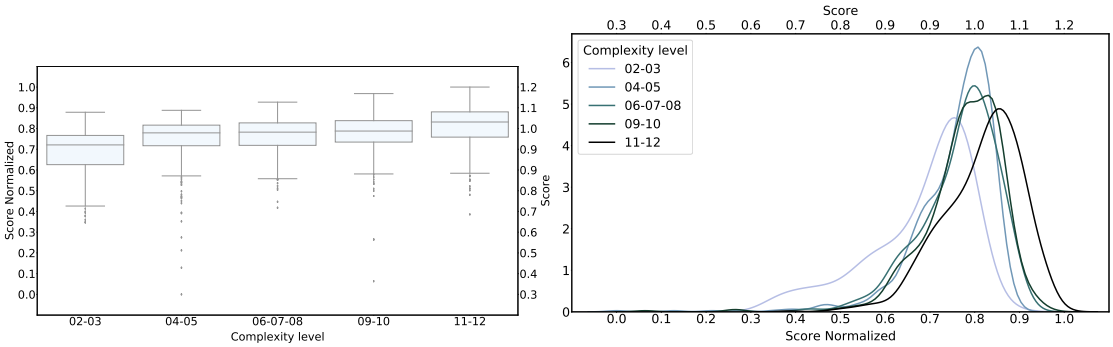
Given this information, even if the results obtained are generally worse compared to the one obtained while validating with Newsela; the models trained on Newsela outperform the results obtained using Mixed, with the Multi-Attentive models showing a slightly better behavior compared to the Multi-Hierarchical ones.



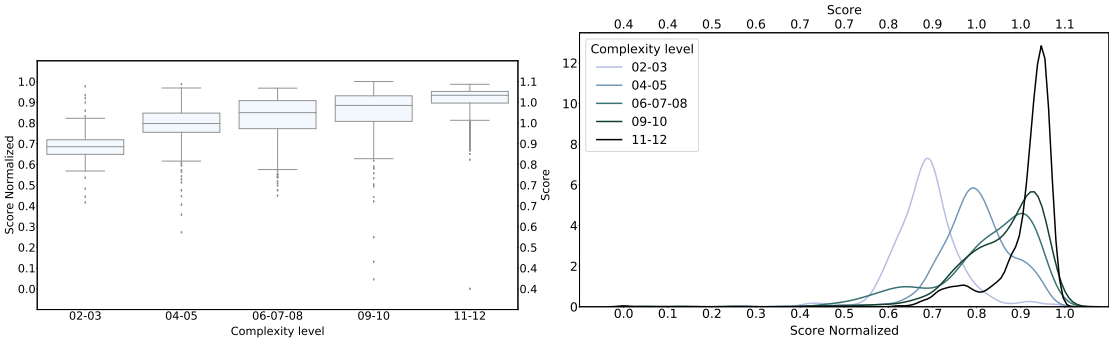
(a) Box plot and distribution using Multi-Attentive model with Mixed for training



(b) Box plot and distribution using Multi-Attentive model with Newsela for training

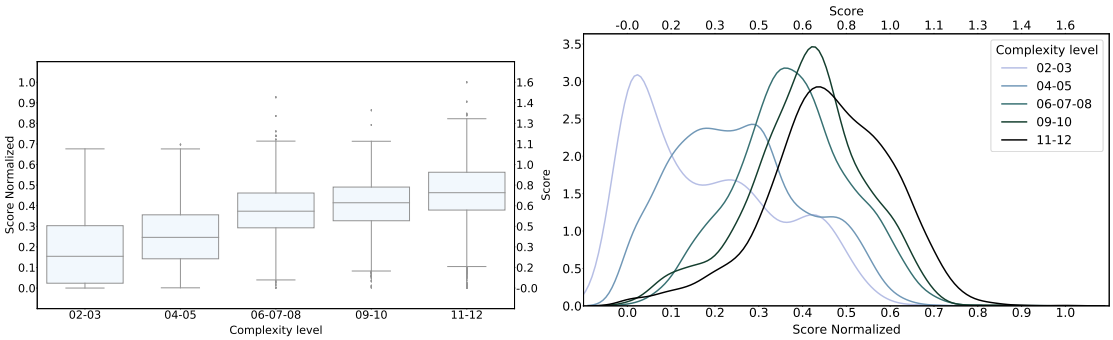


(c) Box plot and distribution using Multi-Hierarchical model with Mixed for training

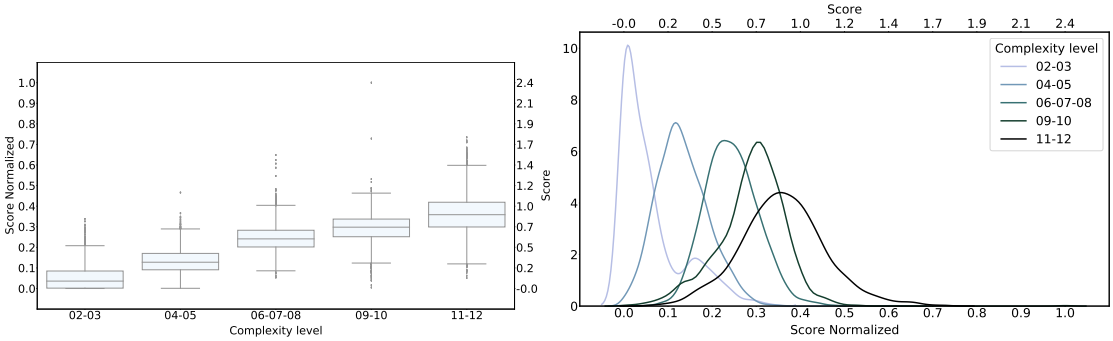


(d) Box plot and distribution using Multi-Hierarchical model with Newsela for training

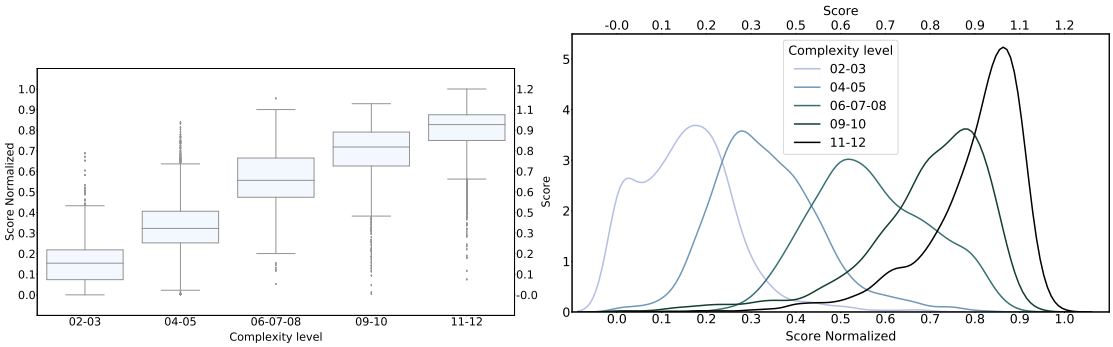
Figure 5.9: Box plot representation, and data distributions of the AppBCCS corpus



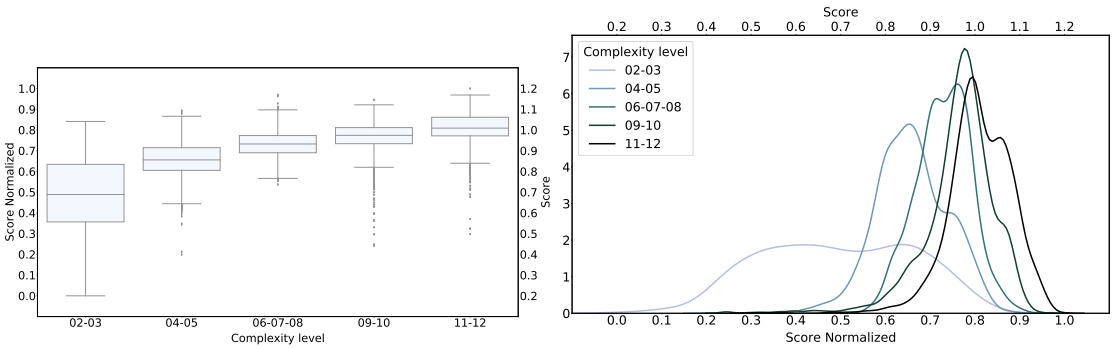
(a) Box plot and distribution using Multi-Attentive model with Mixed for training



(b) Box plot and distribution using Multi-Attentive model with Newsela for training



(c) Box plot and distribution using Multi-Hierarchical model with Mixed for training



(d) Box plot and distribution using Multi-Hierarchical model with Newsela for training

Figure 5.10: Box plot representation, and data distributions of the Mixed corpus

Mixed

The third results are generated by implementing the Mixed corpus, again, given the peculiar nature of this corpus, all the results are obtained as a consequence of a 5-fold approach, granting the absence of shared documents between the training and validation phase.

From these experiments, we expected to obtain an optimal behavior in the approaches based on Mixed and less accurate for the one based on Newsela. While this is respected in the Multi-Hierarchical approach, it is not valid for the Multi-Attentive one, in which the Newsela based approach outperforms the Mixed one.

In particular, the best results are generated by the Multi-Attentive model based on Newsela and the Multi-Hierarchical model based on Mixed, considering these two experiments we can notice that the former is characterized by a smaller variance among levels than the latter.

WeeBit

WeeBit is the fourth dataset considered, and also the first to not be associated with any of the corpus used during the training phase. For this reason, it can be considered as a test of the ability of both models to generalize; however, given the consideration presented while considering the Lexical Complexity, we do not expect particularly good results out of it.

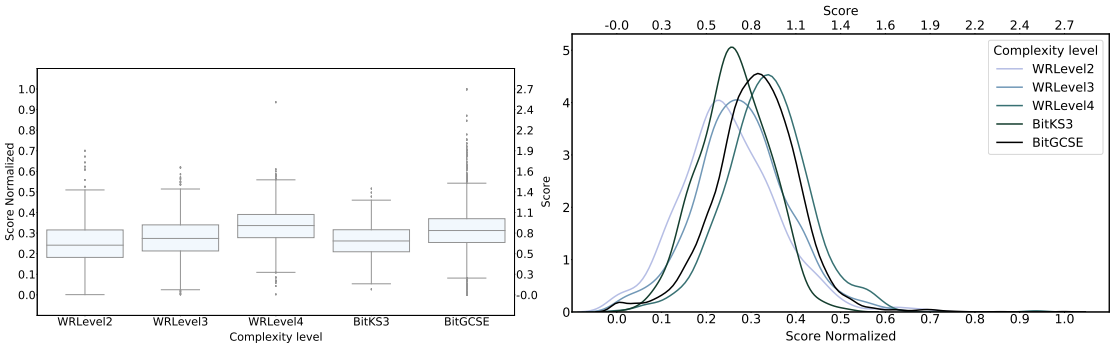
Aligned with the expectation is the results produced by the methods based on the Mixed corpus, characterized by high overlapping and obstruction of some complexity levels, that are not represented in for any score, such as “WRLevel3” and “WRLevel4”.

Surprisingly, instead, the results returned by the approaches based on the Newsela corpus are quite promising, presenting a lower degree of overlapping and allowing every level to be represented in some specific score intervals. Among these two, the most promising result is provided by the Multi-Attentive model that exhibits a lower variance compared to the Multi-Hierarchical one.

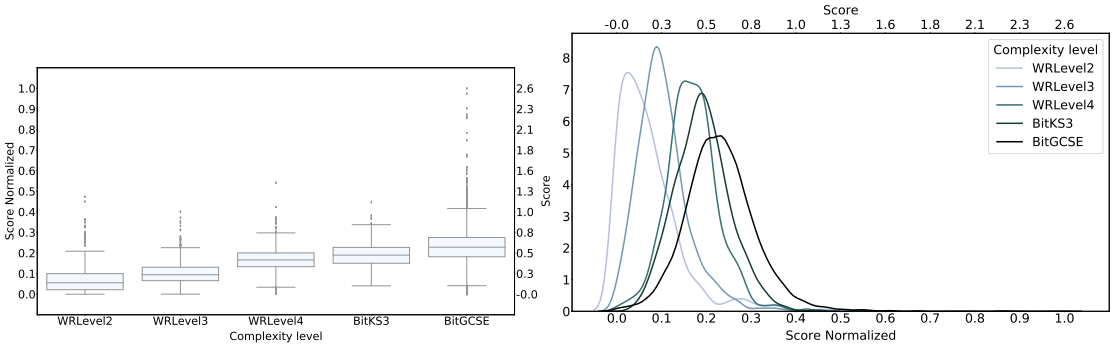
OneStopEnglish

OneStopEnglish is the last dataset analyzed, also with this corpora, as for WeeBit, no relationship is present between the corpora used for the training activity and this one. This corpus can hence act as a second test concerning the capability of our models to generalize.

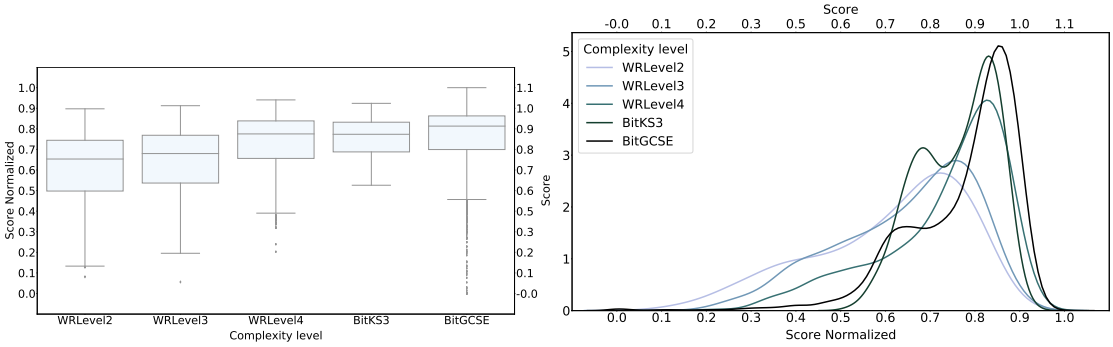
Considering the results, as it happened for the rest of the experiments, it is evident that the approaches based on Newsela perform better than the one based



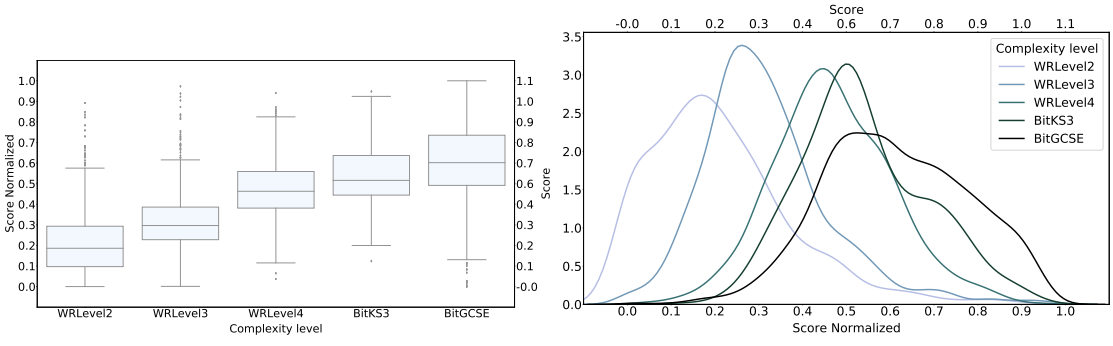
(a) Box plot and distribution using Multi-Attentive model with Mixed for training



(b) Box plot and distribution using Multi-Attentive model with Newsela for training

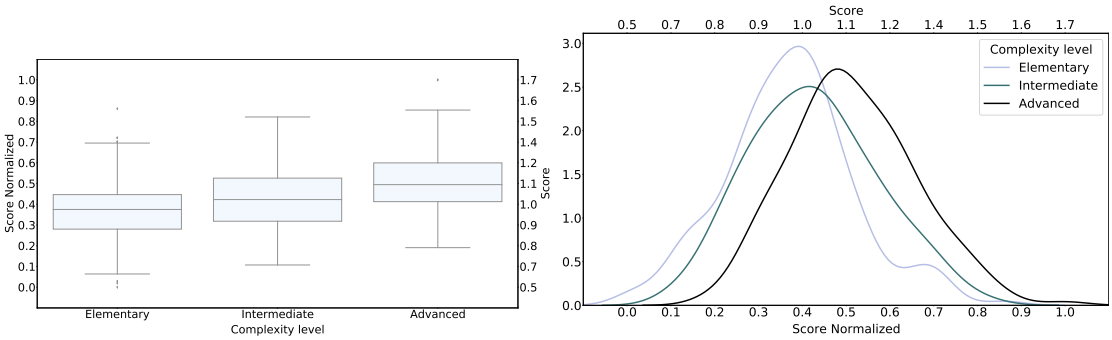


(c) Box plot and distribution using Multi-Hierarchical model with Mixed for training

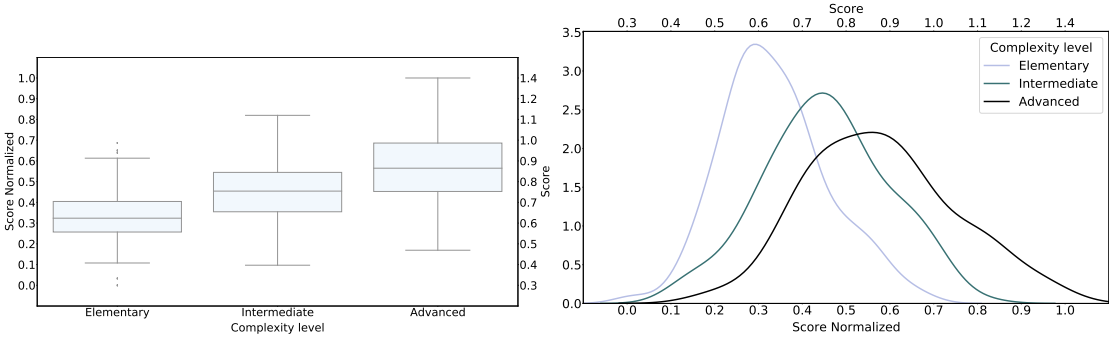


(d) Box plot and distribution using Multi-Hierarchical model with Newsela for training

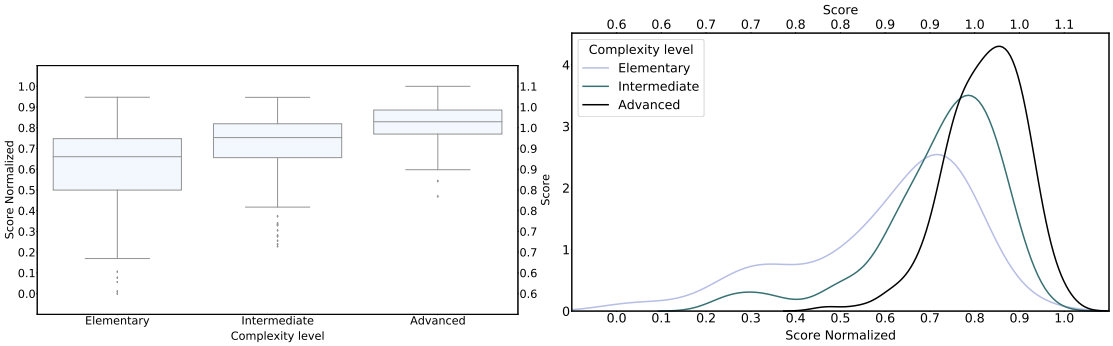
Figure 5.11: Box plot representation, and data distributions of the WeeBit corpus



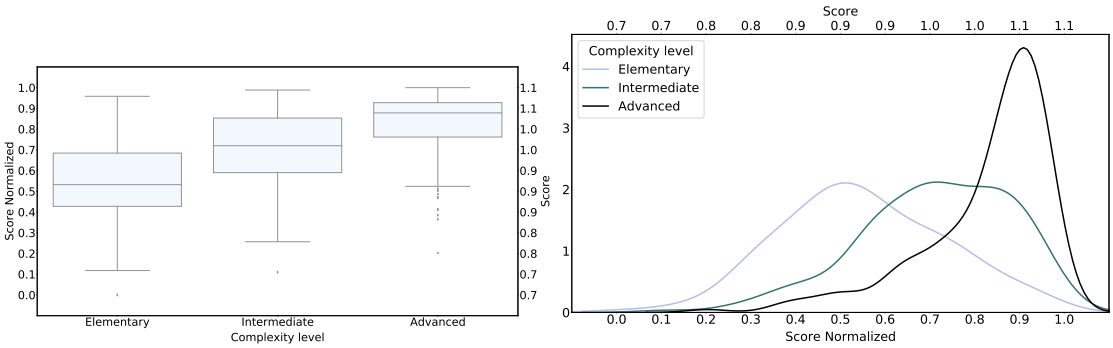
(a) Box plot and distribution using Multi-Attentive model with Mixed for training



(b) Box plot and distribution using Multi-Attentive model with Newsela for training



(c) Box plot and distribution using Multi-Hierarchical model with Mixed for training



(d) Box plot and distribution using Multi-Hierarchical model with Newsela for training

Figure 5.12: Box plot representation, and data distributions of the OneStopEnglish corpus

on Mixed; in particular, the degree of overlapping is inferior and no complexity level is hidden by others, as instead happen for level "Intermediate" in Figure 5.12a.

Lastly, among the two models using Newsela as a training dataset, both are performing quite well. Surprisingly, however, the Multi-Hierarchical model is slightly better, by showing less inter-class variance compared to the other.

5.3.3 Score Validation

In the previous section, we presented the outcome of the score computation step for every corpus and briefly analyzed the various results. Unfortunately, similar to what happened when computing the Lexical Complexity, since these datasets are not classified considering only Syntactic Complexity but a more "general complexity", we can not use the classical validation systems.

What we decided to do, instead, is in line with the approach proposed for the Lexical Complexity, we tested our results by considering the correlation level between our score and a series of low levels metrics introduced in [27].

In this paper, the researchers presented a total of 14 metrics that cover various aspects of Syntactic Complexity and can be grouped in the following categories: Length of Production Unit (3 metrics, that analyze the length of production at the clausal, sentential, or T-unit level), Sentence Complexity (1 metric), Subordination (4 metrics that reflect the amount of subordination), Coordination (3 metrics that measure the amount of coordination), and Particular Structures (3 metrics that consider the relationship between particular syntactic structures and larger production units).

These metrics are the most relevant out of the huge collection of measures initially proposed in [28].

In particular, we evaluated the Pearson correlation coefficient and Spearman rank correlation between our score and such 14 metrics. Tables 5.4 - 5.7 display the complete results, divided in categories.

The meaning of such 14 metrics is briefly presented in the following list:

MLC: Mean Length of Clause

MLS: Mean Length of Sentence

MLT: Mean Length of T-unit

C/S: Sentence Complexity Ratio

C/T: T-unit Complexity Ratio

CT/T: Complex T-uni Ratio

Multi-Attentive - Mixed										
Metric	**AppBCCS**		**Newsela**		**Mixed**		**WeeBit**		**OneStopEnglish**	
	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s
MLC	0.27396	0.28077	0.51890	0.50196	0.48668	0.49607	0.13156	0.15013	0.50102	0.48480
MLS	0.27667	0.30405	0.55166	0.53613	0.59297	0.57619	0.27634	0.29442	0.47134	0.45798
MLT	0.25658	0.28514	0.54795	0.53374	0.59223	0.57536	0.21053	0.23625	0.44206	0.43285
C/S	0.14209	0.15658	0.39302	0.38956	0.46095	0.44732	0.16451	0.19660	0.02994	0.02318
C/T	0.08824	0.09535	0.35661	0.34756	0.45240	0.46035	0.11533	0.13690	0.02064	0.03110
CT/T	0.16135	0.16191	0.31880	0.30616	0.48942	0.46031	0.02372	0.01470	0.03679	0.04026
DC/C	0.15083	0.15286	0.31999	0.30625	0.50961	0.48449	0.04043	0.01953	0.00818	0.02018
DC/T	0.11939	0.14206	0.33609	0.32856	0.48615	0.48245	0.07409	0.06971	0.02144	0.02621
CP/C	0.17803	0.18125	0.40228	0.39130	0.33431	0.36087	0.11653	0.18869	0.36915	0.37166
CP/T	0.18566	0.19654	0.45495	0.44844	0.48360	0.47145	0.17969	0.22910	0.37381	0.38394
T/S	0.19444	0.17381	0.29579	0.30816	0.37271	0.31514	0.14432	0.17094	0.02174	0.01864
CN/C	0.27267	0.29760	0.51698	0.50107	0.53095	0.53625	0.25796	0.27166	0.47062	0.45279
CN/T	0.26317	0.29683	0.53567	0.52421	0.61496	0.58773	0.28810	0.31992	0.41486	0.40648
VP/T	0.12746	0.13175	0.36994	0.36204	0.47536	0.48282	0.03667	0.03395	0.09053	0.10374

Table 5.4: Pearson correlation coefficient (ρ) and Spearman rank correlation (ρ_s) between our score and standard syntactic metrics, per dataset for the Multi-Attentive model based on Mixed corpus

Multi-Attentive - Newsela												
Metric	**AppBCCS**		**Newsela**		**Mixed**		**WeeBit**		**OneStopEnglish**	
	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s
MLC	0.05469	-0.02596	0.75970	0.77585	0.54989	0.68646	0.41329	0.47981	0.61907	0.59981
MLS	0.47809	0.44361	0.93356	0.94388	0.89205	0.88606	0.74890	0.79533	0.91528	0.91293
MLT	0.33153	0.31052	0.91201	0.92488	0.79912	0.83641	0.65029	0.72425	0.86287	0.85436
C/S	0.67227	0.71493	0.77172	0.78670	0.81091	0.77304	0.45263	0.45287	0.41945	0.42473
C/T	0.53845	0.60608	0.70342	0.70870	0.75882	0.71472	0.38701	0.38937	0.42386	0.42244
CT/T	0.53729	0.53057	0.63684	0.62698	0.78071	0.73752	0.44055	0.44739	0.39845	0.40641
DC/C	0.51212	0.48333	0.62486	0.61496	0.76387	0.71946	0.46503	0.46643	0.38126	0.40033
DC/T	0.50446	0.54036	0.66087	0.66515	0.78038	0.72729	0.46919	0.47741	0.41892	0.42411
CP/C	0.06943	-0.05178	0.60125	0.61995	0.41753	0.52855	0.32301	0.40513	0.43301	0.43214
CP/T	0.21371	0.10306	0.73174	0.75226	0.67710	0.68826	0.46606	0.52452	0.58509	0.58458
T/S	0.63497	0.59815	0.56359	0.60026	0.68960	0.61510	0.23704	0.24656	0.02187	0.04281
CN/C	0.06735	0.05069	0.74018	0.74942	0.58053	0.68913	0.45644	0.49219	0.67208	0.63825
CN/T	0.25151	0.23578	0.85257	0.86343	0.79026	0.79714	0.60908	0.65045	0.81056	0.79311
VP/T	0.57472	0.57168	0.74990	0.75679	0.79914	0.76714	0.48626	0.52243	0.53935	0.53700

Table 5.5: Pearson correlation coefficient (ρ) and Spearman rank correlation (ρ_s) between our score and standard syntactic metrics, per dataset for the Multi-Attentive model based on Newsela corpus

Multi-Hierarchical - Mixed										
Metric	**AppBCCS**		**Newsela**		**Mixed**		**WeeBit**		**OneStopEnglish**	
	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s
MLC	0.02082	0.15413	0.40015	0.37874	0.48325	0.59991	-0.03687	0.11665	0.35698	0.43071
MLS	0.24549	0.31380	0.47458	0.45751	0.63591	0.77051	0.23416	0.36765	0.61303	0.70911
MLT	0.17234	0.27795	0.46660	0.45069	0.60677	0.73717	0.15080	0.28641	0.57338	0.67808
C/S	0.34685	0.39173	0.39561	0.38690	0.54627	0.69106	0.27705	0.35514	0.34796	0.36011
C/T	0.28711	0.33152	0.36375	0.35240	0.53307	0.65830	0.19740	0.27495	0.35057	0.37173
CT/T	0.33157	0.33704	0.35745	0.33426	0.62882	0.66360	0.26696	0.18477	0.35013	0.37756
DC/C	0.34927	0.32242	0.35989	0.33476	0.66422	0.69459	0.28226	0.16902	0.37151	0.40442
DC/T	0.28202	0.33630	0.35559	0.34984	0.57126	0.69254	0.22308	0.22871	0.37405	0.40251
CP/C	-0.03778	0.03382	0.29561	0.28583	0.32748	0.45277	0.04871	0.22664	0.24069	0.29979
CP/T	0.04603	0.11053	0.35522	0.35032	0.49061	0.60445	0.14722	0.30018	0.36206	0.43251
T/S	0.34895	0.35627	0.28838	0.29381	0.48895	0.53539	0.18020	0.27177	0.02721	0.00668
CN/C	0.00592	0.15294	0.38629	0.36781	0.53035	0.62831	0.11028	0.27517	0.45075	0.53773
CN/T	0.11341	0.22473	0.43301	0.42348	0.62733	0.72220	0.22315	0.37947	0.57121	0.68260
VP/T	0.32608	0.37486	0.38390	0.37127	0.55738	0.68409	0.16797	0.18187	0.45650	0.46980

Table 5.6: Pearson correlation coefficient (ρ) and Spearman rank correlation (ρ_s) between our score and standard syntactic metrics, per dataset for the Multi-Hierarchical model based on Mixed corpus

Multi-Hierarchical - Newsela												
Metric	**AppBCCS**		**Newsela**		**Mixed**		**WeeBit**		**OneStopEnglish**	
	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s
MLC	-0.17866	-0.00688	0.77911	0.80553	0.47208	0.60116	0.36419	0.45853	0.59443	0.63201
MLS	0.25153	0.40295	0.93910	0.96337	0.57412	0.75252	0.69459	0.78509	0.79135	0.83274
MLT	0.10392	0.29827	0.91674	0.94436	0.54467	0.72060	0.60295	0.70343	0.75723	0.78491
C/S	0.58178	0.71384	0.77231	0.79186	0.50538	0.67178	0.44217	0.46360	0.29709	0.30796
C/T	0.46757	0.60734	0.70018	0.71090	0.49651	0.63639	0.36624	0.39231	0.32062	0.31456
CT/T	0.49547	0.53706	0.63573	0.62840	0.60681	0.64008	0.41880	0.39948	0.29593	0.29444
DC/C	0.53149	0.54446	0.63315	0.62270	0.61980	0.64202	0.47132	0.44157	0.32155	0.32419
DC/T	0.44649	0.58520	0.65888	0.67115	0.51326	0.64951	0.44455	0.46342	0.33408	0.33070
CP/C	-0.22548	-0.11595	0.61508	0.64043	0.32106	0.44759	0.23534	0.34274	0.38667	0.41011
CP/T	-0.08544	0.03508	0.73724	0.77113	0.44678	0.58089	0.38049	0.46277	0.49738	0.52003
T/S	0.55514	0.61508	0.57757	0.61221	0.47948	0.53255	0.23059	0.26256	-0.02561	0.02681
CN/C	-0.16656	0.07915	0.75400	0.77560	0.48364	0.59385	0.41235	0.46029	0.63938	0.65154
CN/T	0.00835	0.24509	0.85381	0.88298	0.56092	0.68162	0.56438	0.62202	0.72154	0.74571
VP/T	0.45067	0.54781	0.75511	0.77020	0.53095	0.67555	0.46471	0.51442	0.43718	0.43429

Table 5.7: Pearson correlation coefficient (ρ) and Spearman rank correlation (ρ_s) between our score and standard syntactic metrics, per dataset for the Multi-Hierarchical model based on Newsela corpus

DC/C: Dependent Clause Ratio

DC/T: Dependent Clauses per T-unit

CP/C: Coordinate Phrases per Clause

CP/T: Coordinate Phrases per T-unit

T/S: Sentence Coordination Ratio

CN/C: Complex Nominals per Clause

CN/T: Complex Nominals per T-unit

VP/T: Verb Phrases per T-unit

Similar to the approach implemented while evaluating the quality of the predicted Lexical Complexity, we evaluated the correlation strength by implementing Cohen's interpretation. In particular, if the correlation is lower then 0.10 there is no association in any form; a correlation between 0.10 and 0.30 indicates a small association; a correlation between 0.30 and 0.50 represents a medium association, and lastly, a correlation higher than 0.50 implies a large association. Table 5.8 presents the result of this correlation.

If our assumptions presented in the previous section are correct, from this results, we expect a better performance by the two structure that features Newsela as the training dataset, with the Multi-Attentive model coming out as the winner among all the implemented variants.

Looking at the table, this is exactly what happens, with the Multi-Attentive model based on Newsela outperforming the methods based on Mixed and begin slightly better than the Multi-Hierarchical model based on Newsela.

It is relevant to notice that the best model is also the only one that for every corpus and every metrics shows a medium or large correlation for 13 metrics out of 14; besides the metric with a null or small association in WeeBit and OneStopEnglish is the Sentence Coordination Ratio (T/S) that shows poor compatibility with the two corpora in every experiment.

Lastly, the only other metric showing a small association is Coordinate Phrases per Clause (CP/C), which generally perform badly with AppBCCS for all approaches.

Interesting to notice is that both metrics belong to the Coordination Group, proving that out of all the five groups, these metrics are the one less compatible with our score.

Summing up, our score seems to correlate with almost all the low-level metrics, being able to "concentrate" in just one value the semantics carried by multiple measures.

Table 5.8: Total correlation strength of the various approaches

Strength	AppBCCS		Newsela		Mixed		WeeBit		OneStopEnglish	
	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s	ρ	ρ_s
Multi-Attentive - Mixed										
Large	0	0	5	5	5	4	0	0	1	0
Medium	9	9	8	9	9	10	0	1	6	7
Small	5	5	1	0	0	0	10	9	0	1
None	0	0	0	0	0	0	4	4	7	6
Multi-Attentive - Newsela										
Large	11	11	14	14	13	14	3	5	7	7
Medium	2	3	0	0	1	0	10	8	6	6
Small	1	0	0	0	0	0	1	1	0	0
None	0	0	0	0	0	0	0	0	1	1
Multi-Hierarchical - Mixed										
Large	0	3	0	0	10	13	0	0	3	4
Medium	10	10	12	12	4	1	0	4	9	8
Small	4	1	2	2	0	0	12	10	1	1
None	0	0	0	0	0	0	2	0	1	1
Multi-Hierarchical - Newsela										
Large	3	7	14	14	8	13	3	4	5	6
Medium	4	1	0	0	6	1	9	9	6	6
Small	2	2	0	0	0	0	2	1	2	1
None	5	4	0	0	0	0	0	0	1	1

5.3.4 Scale Generation

The last step to complete a full twin procedure with the Lexical Complexity analysis is to introduce a scale system that gives a semantic to the values returned by our model; in particular, in this section, we are going to propose the methodology and a possible application of it.

The first step to generate a scale system is to select one of the approaches and a specific corpus to use as a base. Considering the results reported by the methods, we can rank them, from first to last as:

- Multi-Attentive model trained on Newsela
- Multi-Hierarchical model trained on Newsela
- Multi-Hierarchical model trained on Mixed

- Multi-Attentive model trained on Mixed

This ranking is a direct consequence of the better behavior introduced by the usage of Newsela compared to Mixed as base corpus during the training. We believe that the origin of this phenomenon is intrinsic to the corpora themselves. Mixed is, in fact, the result of the union of Newsela and AppBCCS corpora, allowing for a broader range of genres compared to the simple articles proposed in Newsela. Such diversity, however, is in place only by name.

If we consider the results presented in Section 3.2.3, it is evident that the text samples belonging to Newsela dominate the corpus, and the influence of AppBCCS is limited to only a small portion. Such a share, being too small, generate the opposite effects by strongly increasing the variance in the dataset, hence increasing also the number of text samples perceived as outliers by the network, and as consequence, worsening the learning capability of the network.

The approaches based on Newsela, on the other side, feature a lower diversity in the text samples allowing the network models to learn better the features that characterize every complexity level.

Among the two best performing approaches, the Multi-Attentive model comes out as the winner, considering both the results of the correlation analysis and the distribution of the scores (see Section 5.3).

After selecting the model, we need to decide which corpus use. Among the five datasets tested using the Multi-Attentive model, the two best performing are Newsela and Mixed.

In this step, we are looking for a corpus that provides the highest complexity level diversification while maintaining a lower intra-level variance. The first condition is easily satisfied since both corpus are divided in the same number of levels while for the second we can look at the boxplot presented in Section 5.3.2, here reported in Figure 5.13 for convenience.

Considering the figure, Mixed appears to be the corpus with the lowest intra-variance on the complexity level; however, if we pay closer attention, the lower

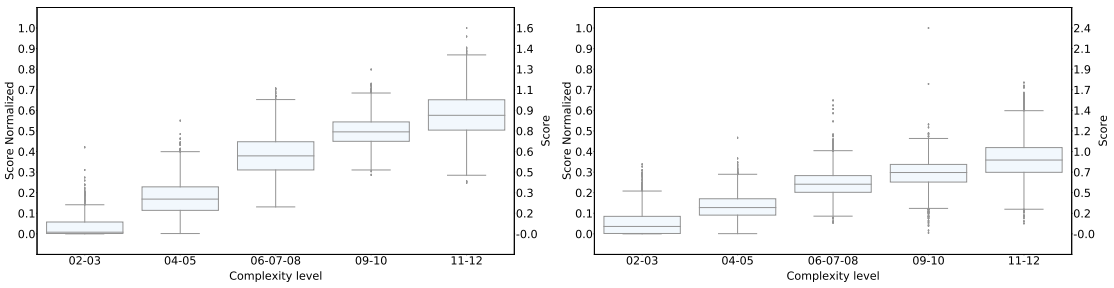


Figure 5.13: Comparison between the box plot representation of score for Newsela (left) and Mixed (right)

Table 5.9: Table showing the distributions that best fit data

Comp. level c	Distribution $f_c(s)$	Notes
"02-03"	Normal($\mu = 0$, $\sigma = 0.02023581$)	Prob. mass is 50%
"04-05"	Normal($\mu = 0.17018700$, $\sigma = 0.07747836$)	Prob. mass is 98.5%
"06-07-08"	Normal($\mu = 0.38106293$, $\sigma = 0.09353859$)	
"09-10"	Normal($\mu = 0.49465190$, $\sigma = 0.06711004$)	
"11-12"	Normal($\mu = 0.57584023$, $\sigma = 0.10797352$)	

variance is the result of the huge increment in the number of outliers.

This phenomenon, as mentioned above, is the consequence of the structure of Mixed, which has more data then Newsela but more sparse. While this aspect might seem irrelevant at first impact, it becomes a major hindrance during the definition of a scale, because all the outliers are going to be removed during the process.

The best outcome would be to have a dataset with a variety of content similar to Mixed, but with a higher amount of text samples, that assert the relevance of every genre.

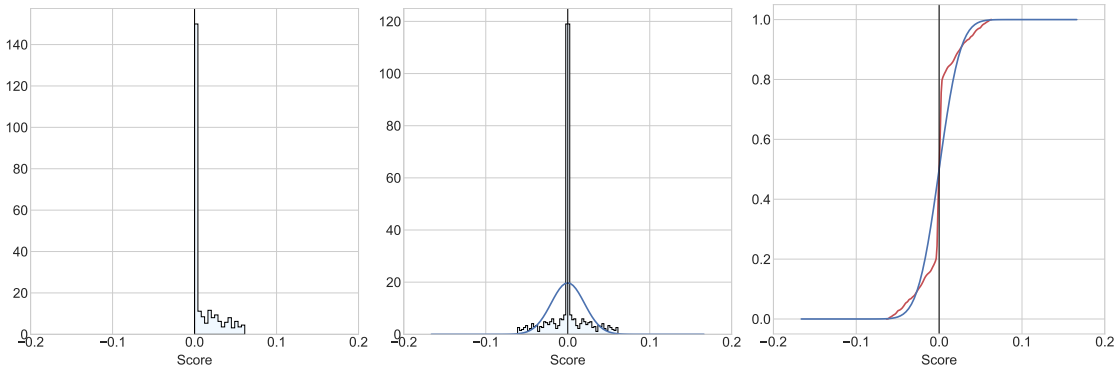
Given this premise, we believe that Newsela can produce results with a higher level of reliability compared to Mixed, hence we will use the former. We expect, however, to encounter some problems while proceeding with the scale generation, in particular, within the levels with fewer data.

Once the dataset is selected, we can proceed with the next steps of the procedure: the analysis of the distributions of scores. This phase is very similar if not identical to the one executed for the Lexical Complexity, hence we are going to cover it briefly remanding to Section 4.3 for any specific detail.

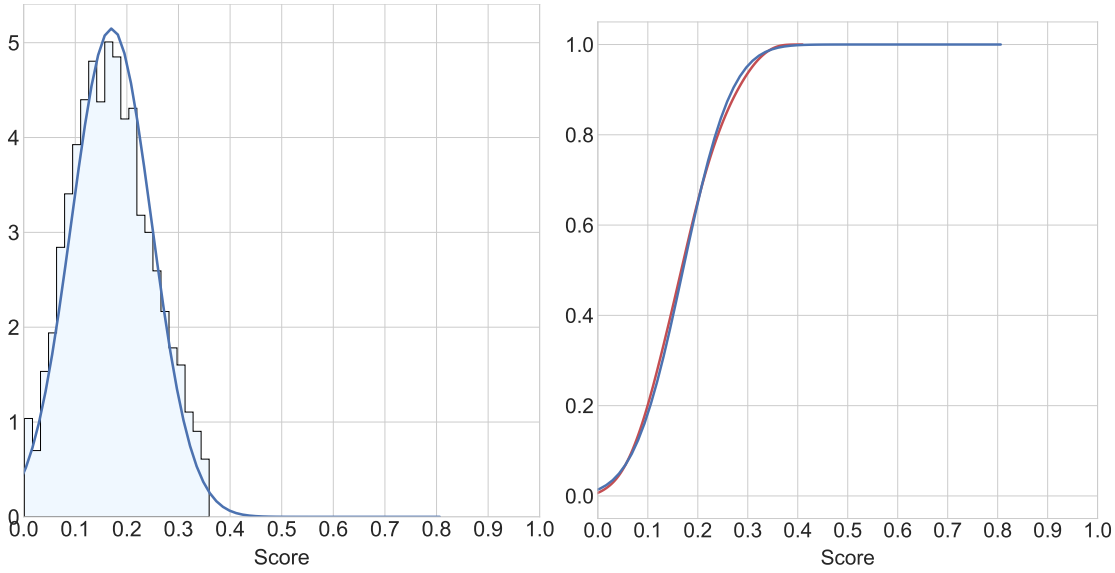
After removing the outliers applying the medcouple rules, we can define $f_c(x)$ as the density function (DF) of a probability distribution approximating the distribution of the score s for the complexity level c, in the "cleaned" corpus. To find $f_c(s)$, we used the Kolmogorov-Smirnov test that pointed out the most promising functions. Then, we selected the best fitting function by comparing the behavior of a candidate $f_c(s)$ and its Cumulative Density Function (CDF), against the DF and CDF reconstructed from the corpus data.

Figure 5.14 illustrates the results of this procedure, where DF of the corpus data is represented as a histogram, while its CDF is drawn as a curve; notice that the CDF reconstructed from the corpus data and the candidate distribution CDF are nearly indistinguishable. Table 5.9 displays the parameters of the identified $f_c(s)$.

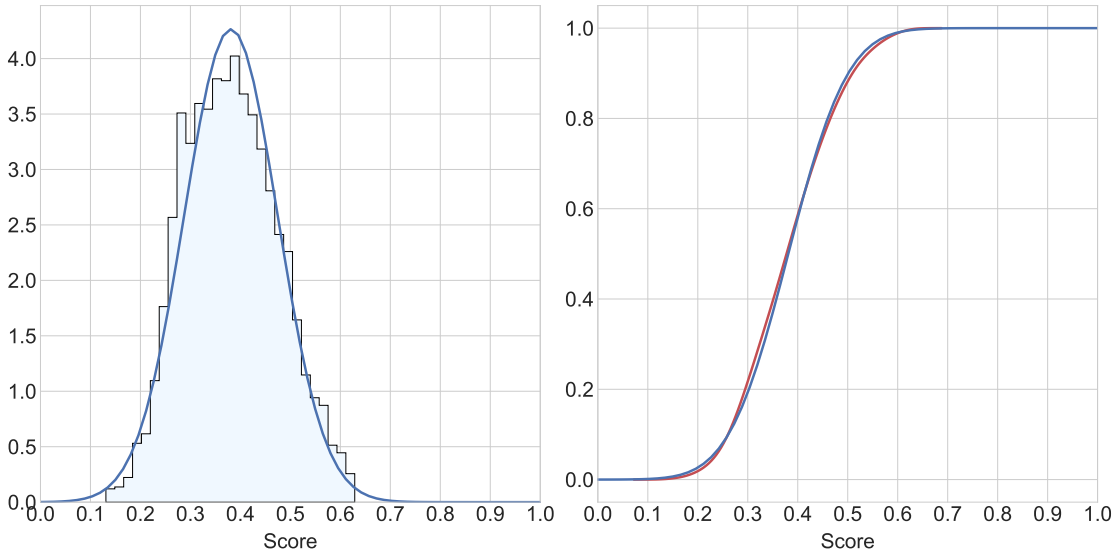
For level "02-03", we suspected a similar behavior to what happened in the definition of a scale for Lexical Complexity, with the data depicting half of the



(a) True histogram, DF and CDF of level "02-03"

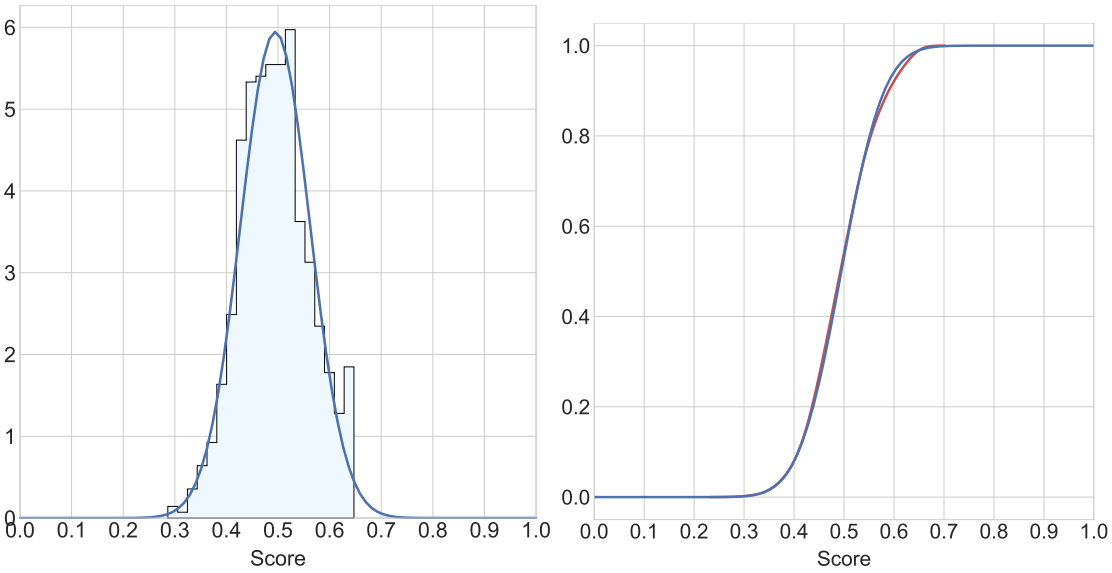


(b) DF and CDF of level "04-05"

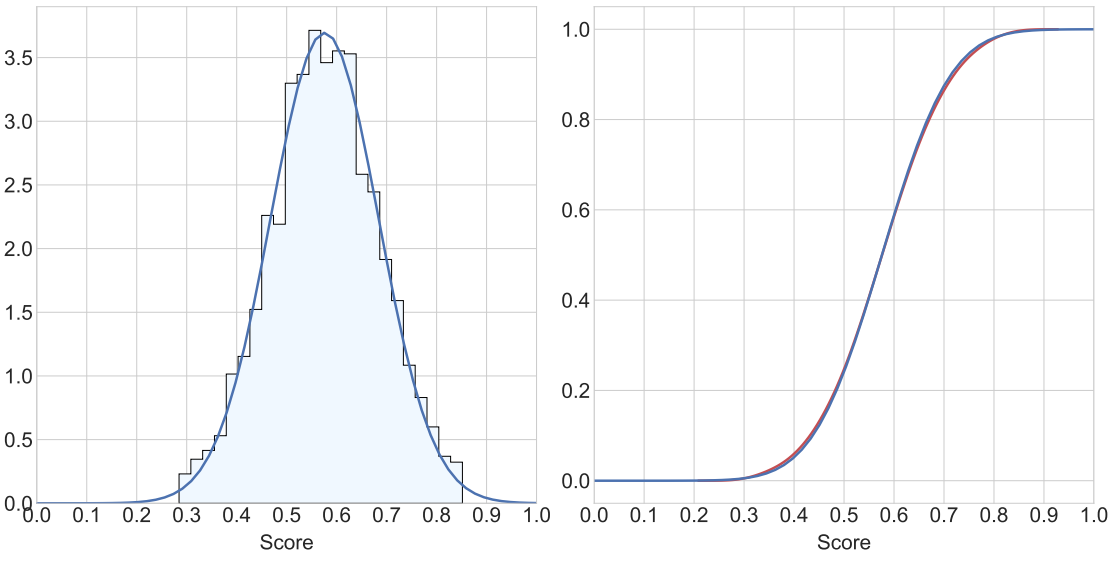


(c) DF and CDF of level "06-07-08"

Figure 5.14: For every complexity level (Newsela corpus): histogram of the data DF; distribution $f_c(s)$; CDF for both data and distribution.



(d) DF and CDF of level "09-10"



(e) DF and CDF of level "11-12"

Figure 5.14: For every complexity level (Newsela corpus): histogram of the data DF; distribution $f_c(s)$; CDF for both data and distribution.

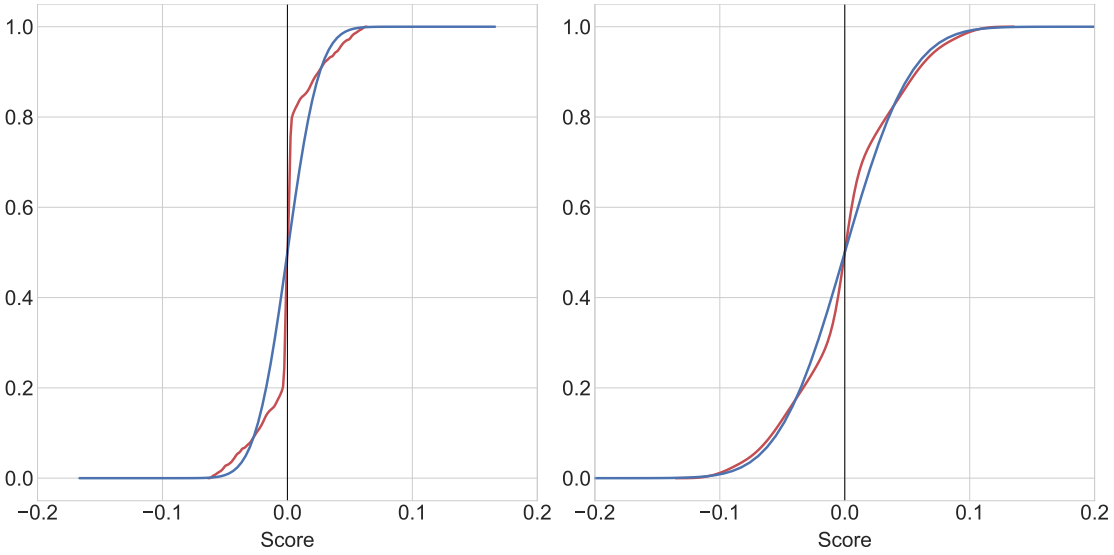


Figure 5.15: Comparison between CDF representation of level "02-03" for Newsela (left) and Mixed (right)

Normal distribution. However, when we implemented the same approach and mirrored the data around zero, we noticed that some misalignment was present between the identified CDF and the one generated by the data. (see Figure 5.14a)

Such error is in line with what we expected and is, in our opinion, a direct influence of the low amount of data that characterize such level. To understand if the decision of using a normal as identification function was correct, we analyzed the effect on level "02-03" for the Mixed corpus.

Figure 5.15 displays this comparison and highlight how increasing the number of data available defines a distribution similar to a Normal. For this reason, even if the approximation we presented has a certain degree of error, such approximation is probably correct since increasing the amount of data will lead to the identification of a Normal distribution.

Similarly to what happened in class "04-05" while considering the Lexical Complexity, the Normal DF accounts for 98.5% of the probability mass (as part of the mass is distributed to $s < 0$), given the small nature of the error, we decided to ignore it.

Then, being $f_c(s)$ a continuous distribution, the likelihood that an interval centered in s, with radius δ, belongs to the complexity level c is

$$\begin{aligned} L(s-\delta < x < s+\delta | c) &= \left(1 + \mathbb{1}_{\{\text{“02-03”}\}}(c)\right) \cdot \int_{s-\delta}^{s+\delta} f_c(x)dx \\ \forall c \in S_c &= \{\text{“02-03”}, \text{“04-05”}, \text{“06-07-08”}, \text{“09-10”}, \text{“11-12”}\} \end{aligned} \tag{5.12}$$

where S_c is a totally ordered set (and the order relation is obviously defined) and $\mathbb{1}_{\{\text{“02-03”}\}}(c)$ is the indicator function. We can then define $K(s)$, a function that

associates a given score s to a complexity level, $K(s) : [0, 1] \subset \mathbb{R} \to S_c$; then, the probability of s belonging to a complexity level c is

$$P_\delta(K(s) = c) = \frac{L(s - \delta < x < s + \delta | c)}{\sum_{c' \in S_c} L(s - \delta < x < s + \delta | c')} \tag{5.13}$$

and, if authors are required to produce texts with complexity level no higher than $\tilde{c}$, the probability that the complexity level associate to a given score is not higher than $\tilde{c}$, is

$$P_\delta(K(s) \leq \tilde{c}) = \sum_{c' \leq \tilde{c} \in S_c} P_\delta(k(s) = c'). \tag{5.14}$$

Figure 5.16, on top, shows graphs of $P_\delta(K(s) = c)$; then, we can define our scale calculating the most likely complexity level of score s, as

$$\hat{c} = \arg\max_{c \in S_c} P(K(s) = c) = \arg\max_{c \in S_c} L(s - \delta < x < s + \delta | c). \tag{5.15}$$

Given the thresholds $[s_0 = 0, \ldots, s_5 = 1]$ for s, we can calculate s' taking values in a scale with equidistant thresholds $[s'_0 = 0, s'_1 = 0.2, s'_2 = 0.4, s'_3 = 0.6, s'_4 = 0.8, s'_5 = 1]$, maybe more convenient (although the complexity "density" changes in each level of the scale). Given s and s', and being (s_{i-1}, s_i), (s'_{i-1}, s'_i) the borders of the complexity level they belong to (of course, s and s' belong to the same complexity level), the relationship between the two scales is

$$\frac{s_i - s_{i-1}}{s'_i - s'_{i-1}} = \frac{s - s_{i-1}}{s' - s'_{i-1}}. \tag{5.16}$$

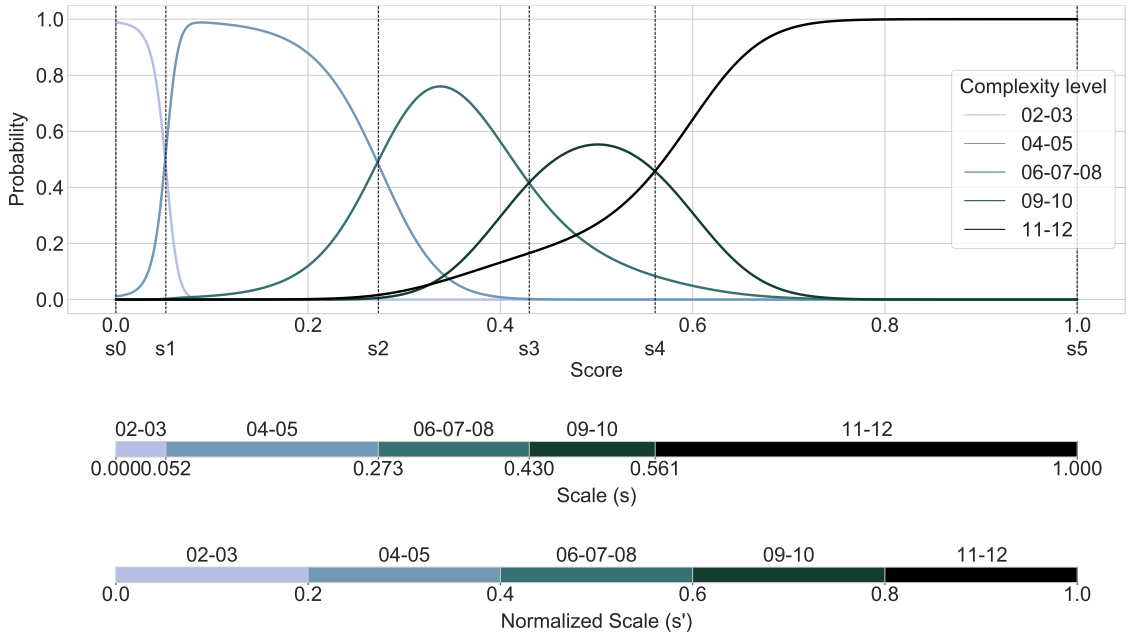


Figure 5.16: Probability that the score s belongs to a complexity level ($\delta = 10^{-6}$); scales for s and s'

A visual representation of the two scales, for s and s', is provided on the lower part of Figure 5.16.

Also in this case, as for the Lexical Complexity, the approach presented here is based on the approximation made on the behavior of distribution "04-05", however, if such approximation is considered unacceptable then the equation can be proposed in a generalized fashion as:

$$L(s-\delta < x < s+\delta|c) = \left(1 + \sum_{c'} \frac{e_{c'}}{1-e_{c'}} \mathbb{1}_{\{c'\}}(c)\right) \cdot \int_{s-\delta}^{s+\delta} f_c(x)dx \tag{5.17}$$

where, for each complexity level c', $e_{c'}$ is the fraction of the $f_{c'}(s)$ probability mass that does not belong to the $[0,1] \subset \mathbb{R}$ range.

It is easy to show that Equation 5.17 is still a CDF in $[0,1] \subset \mathbb{R}$:

$$\left(1 + \sum_{c'} \frac{e_{c'}}{1-e_{c'}} \mathbb{1}_{\{c'\}}(c)\right) \cdot \int_{0}^{1} f_c(x)dx = 1 \tag{5.18}$$

In particular, considering $c' \in \{$"02-03", "04-05"$\}$ and $e_{c'} \in \{0.5, 0.015\}$ we can improve the approximation provided by Equation 5.12 and thus the probability provided by Equation 5.13.

Chapter 6

Conclusion

6.1 Conclusions

In this thesis, we proposed BASILISCo, a novel approach to compute the complexity of a document. More specifically, a system based on the distinct computation of lexical and reading complexity was designed, tested, and implemented so to better assess the problem, compared to the current state of the art solutions.

Literature research demonstrated that the problem of identifying specifics aspects of the complexity of documents underwent lower attention. In the recent works, characterized by a congregation of information, the point of focus has always been a general reading complexity, depriving the user of a better understanding of the features associated with such complexity.

To avoid such limitations, starting from the current state of the art, we identified and reinvented multiple designs to address our specific task.

This leads to the definition of BASILISCo to study multiple aspects of complexity, in particular, we analyzed the complexity deriving from lexical and syntactic features. Both approaches are completely independent, however, both are implemented following the same workflow that can be summed up in the following six steps: feature identification or selection, model generation, score computation, score validation, score distribution analysis, and scale generation. Using this workflow allowed us to produce results that are characterized by a strong consistency, independently by the nature of the lower process effectively implemented.

Lexical Complexity was implemented reinventing the log-likelihood ratio test, an approach originally proposed for unsupervised content selection. Syntactic Complexity was instead computed using state of the art techniques and innovative models of deep learning generated by redesigning models normally used in natural language processing tasks.

Given the innovative nature of the obtained results, a comparison via classical

metrics with predefined datasets was not possible since corpora that measure only specific aspects of the complexity do not exist. Hence, we evaluated the correctness of the approach by mean of correlation analysis with specific low levels metrics responsible to grasp different aspects of lexical and Syntactic Complexity. These tests, of which results are reported in their summed up version in Tables 4.3 and 5.8, proved that our models strongly correlates with the majority of the features, implying the quality of the returned score, and their validity in defining the complexity of documents.

Lastly, we proposed a methodology for defining a scale able to provide meaning to our score by associating it to specific complexity levels, for which we presented also examples of implementation. Such examples are based on assumptions and ultimately limited by the low amount of data available through the corpora; however, they proved to be valid and acceptable in the current context.

Finally, a clarification is due: BASILISCo is based on the assumption that the reader possesses a general knowledge of the topics covered by the analyzed documents. That assumption holds for the datasets we relied upon, as they are typical of a school environment, which represents the general knowledge required in daily life and thus can be defined as "common knowledge". Some adjustments should be done if documents treat arguments belonging to specific fields, like the medical one. A solution would be either to use a tuned corpus to define the scale or to provide a separate score considering independently, people with or without knowledge about that argument.

6.2 Future works

The implemented approaches represent a first step in the necessity of providing the user with complete and detailed information concerning the complexity of text documents; however, multiple improvements can be done, either increasing the level of precision of the reported information or proving the validity of the returned score.

For example, a possible improvement would be to extend the current scope of analysis by introducing an investigation on semantic, not covered by the current research, but that contributes to the reading complexity of a document. The implementation will feature different low-level processes to produce the results, but thanks to the definition of a general workflow, the newly generated results will correlate with ours.

A different improvement direction would instead be to increase the robustness to different aspects of lexical and Syntactic Complexity that might be not covered in this research or that are dependent by the reader itself, and as such not considered.

Finally, given the nature and aim of the task, to completely prove the validity of the approach, an experiment involving human readers is needed. In this way, it will be possible to verify if the returned complexity and the one identified by the readers correlate.

Appendix A

Composition of the custom version of Appendix-B Common Core Standard corpus (AppBCCS)

In this appendix we present the composition of our version of the AppBCCS corpus. The table is grouped firstly by complexity level, then by genre and lastly by author. For every author we highlight the number of lemmas, tokens and chapters. This representation aims at showing the relevance of every author and genre for a specific complexity level.

Table 6.1: Composition of our version of the AppBCCS corpus

Gold	Genre	Author	N. Lemma	N. Words	N. Chapters
"02-03"	Informational Texts	AA	114	329	1
		AB	76	279	1
		AC	107	294	1
		AD	74	276	1
		AE	66	208	1
		AF	99	269	1
	Stories	AG	90	567	1
		AH	1826	23845	48
		AI	810	7545	10
		AJ	36	106	1
		AK	91	291	1
		AL	2386	42985	60
		AM	1777	51850	58

Table 6.1: Composition of our version of the AppBCCS corpus

Gold	Genre	Author	N. Lemma	N. Words	N. Chapters
"04-05"	Informational Texts	AN	47	115	1
		AO	43	183	1
		AP	89	251	1
		AQ	163	602	1
		AR	92	227	1
		AS	60	194	1
		AT	85	282	1
		AU	59	237	1
		AV	124	380	1
		AW	76	273	1
		AX	99	373	2
	Stories	AY	1465	16865	27
		AZ	3213	80539	27
		BA	1708	26527	12
		BB	3620	99564	40
		BC	89	298	1
		BD	2585	42663	48
		BE	2425	27643	27
		BF	471	2120	1
		BG	3653	54413	17
		BH	4728	99492	43
		BI	3202	68648	14
		BJ	11334	1050456	400

Table 6.1: Composition of our version of the AppBCCS corpus

Gold	Genre	Author	N. Lemma	N. Words	N. Chapters
"06-07-08"	Informational Texts: English Language Arts	BK	71	255	1
		BL	4040	31912	11
		BM	5795	75072	4
		BN	3449	53329	22
	Informational Texts: History and Social Studies	BO	41	98	1
		BP	3745	52178	10
	Informational Texts: Science, Mathematics, and Technical Subjects	BQ	123	481	1
		BR	67	187	1
		BS	3347	34897	86
		BT	1900	28826	13
		BU	133	363	1
		BV	111	268	1
	Stories	BW	12083	174130	47
		BX	3217	49160	12
		BY	3719	46221	22
		BZ	8393	343447	75
		CA	6756	66190	35

Table 6.1: Composition of our version of the AppBCCS corpus

Gold	Genre	Author	N. Lemma	N. Words	N. Chapters
"09-10"	Drama	CB	1680	26205	3
		CC	2221	17462	28
		CD	3372	37123	1
	Informational Texts: English Language Arts	CE	6018	81760	37
		CF	1308	8727	1
		CG	222	725	1
		CH	110	1646	1
		CI	741	3338	1
		CJ	364	1567	1
		CK	1316	6157	1
	Informational Texts: History and Social Studies	CL	9007	137719	11
		CM	9262	194802	26
		CN	5309	151384	20
		CO	4767	61002	17
	Informational Texts: Science, Mathematics, and Technical Subjects	CP	3628	51661	8
		CQ	5282	97624	4
		CR	85	318	1
	Stories	CS	4966	95082	21
		CT	2988	51228	25
		CU	3009	32683	30
		CV	1857	22006	3
		CW	618	2743	2
		CX	5301	99251	31
		CY	511	2070	1
		CZ	5184	108487	24
		DA	5548	179220	30
		DB	4465	106733	23
		DC	7563	250596	111
		DD	3466	46038	3
		DE	5626	79297	28

Table 6.1: Composition of our version of the AppBCCS corpus

Gold	Genre	Author	N. Lemma	N. Words	N. Chapters
"11-12"	Drama	DF	2184	17544	5
		DG	1791	20472	3
		DH	3217	31284	5
	Informational Texts: English Language Arts	DI	4540	59602	35
		DJ	3412	28313	6
		DK	6882	106697	18
		DL	744	2912	1
		DM	1767	13819	9
		DN	2392	21110	5
	Informational Texts: History and Social Studies	DO	7262	307742	119
		DP	303	999	1
		DQ	1914	10478	1
	Informational Texts: Science, Mathematics, and Technical Subjects	DR	547	3719	1
		DS	3924	43345	5
		DT	4794	72589	9
		DU	79	201	1
		DV	666	2840	1
		DW	534	3202	1
		DX	853	3994	1
	Stories	DY	922	4898	3
		DZ	8533	185365	38
		EA	595	2314	1
		EB	3284	88862	41
		EC	4096	48379	9
		ED	6505	203832	41
		EE	4319	30620	30
		EF	4277	120916	61
		EG	6256	103536	12
		EH	949	4249	1
		EI	8940	400671	116
		EJ	5258	68135	24
		EK	844	4263	2
		EL	10749	249809	26
		EM	4365	52248	5
		EN	2740	57246	1
		EO	3436	59854	20

Analyzing the table we can notice some characteristics. Firstly, Stories represents the genre with the highest influence in the dataset, being the one with the highest number of authors and highest values in terms of lemmas, words and chapters. Secondly, Stories accounts for the highest diversification in terms of chapters (and, consequently, weight in the distributions) compared to any other genre. Finally, the influence of some authors is much higher than others; this can be easily seen by looking at the amount of chapters and unique lemmas in their works.

Bibliography

[1] O. De Clercq and V. Hoste, "All mixed up? finding the optimal feature set for general readability prediction and its application to english and dutch," *COMPUTATIONAL LINGUISTICS*, vol. 42, no. 3, pp. 457–490, 2016. [Online]. Available: http://dx.doi.org/10.1162/COLI_a_00255

[2] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *ArXiv*, vol. 1409, 09 2014.

[3] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," *CoRR*, vol. abs/1601.06733, 2016. [Online]. Available: http://arxiv.org/abs/1601.06733

[4] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[5] I. M. Azpiazu and M. S. Pera, "Multiattentive recurrent neural network architecture for multilingual readability assessment," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 421–436, 2019.

[6] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 01 2016, pp. 1480–1489.

[7] W. Dubay, "Unlocking language: The classic readability studies," *Professional Communication, IEEE Transactions on*, vol. 51, pp. 416 – 417, 01 2009.

[8] M. Heilman, K. Collins-Thompson, and M. Eskenazi, "An analysis of statistical models and features for reading difficulty prediction," in *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*, ser. EANL '08. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008, pp. 71–79. [Online]. Available: http://dl.acm.org/citation.cfm?id=1631836.1631845

[9] F. Rudolph, "A new readability yardstick," *Journal of Applied Psychology*, vol. 32(3), 221-233, 1948. [Online]. Available: http://dx.doi.org/10.1037/h0057532

[10] M. Proust, *Swann's Way*. Simon & Brown, 2018.

[11] K. J. Peter, R. P. Fishburne, R. L. Rogers, and B. S. Chissom, "Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel," *Institute for Simulation and Training*, 1975. [Online]. Available: https://stars.library.ucf.edu/istlibrary/56

[12] J. S. J. S. Chall and E. Dale, *Readability revisited : the new Dale-Chall readability formula*. Cambridge, Mass. : Brookline Books, 1995, includes bibliographical references (p. [151]-155) and index.

[13] L. Si and J. Callan, "A statistical model for scientific readability," in *Proceedings of the 10th International Conference on Information Knowledge Management (ICKM-2001)*, 2001.

[14] K. Collins-Thompson and J. P. Callan, "A language modeling approach to predicting reading difficulty," in *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*. Boston, Massachusetts, USA: Association for Computational Linguistics, May 2 - May 7 2004, pp. 193–200. [Online]. Available: https://www.aclweb.org/anthology/N04-1025

[15] K. Collins-Thompson, "Computational assessment of text readability: A survey of current and future research," *ITL - International Journal of Applied Linguistics*, vol. 165, pp. 97–135, 01 2014.

[16] M. Heilman, K. Collins-Thompson, J. Callan, and M. Eskenazi, "Combining lexical and grammatical features to improve readability measures for first and second language texts." in *Proceedings of the Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL 2007)*, 01 2007, pp. 460–467.

[17] S. E. Schwarm and M. Ostendorf, "Reading level assessment using support vector machines and statistical language models," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2005, pp. 523–530.

[18] T. Landauer, K. Kireyev, and C. Panaccione, "Word maturity: A new metric for word knowledge," *Scientific Studies of Reading*, vol. 15, pp. 92–108, 01 2011.

[19] K. Kireyev and T. Landauer, "Word maturity: Computational modeling of word knowledge," in *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 01 2011, pp. 299–308.

[20] X. Chen and D. Meurers, "Word frequency and readability: Predicting the text-level readability with a lexical-level attribute," *Journal of Research in Reading*, vol. 41, no. 3, pp. 486–510, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/1467-9817.12121

[21] N. Nouri and B. Zerhouni, "Lexical frequency effect on reading comprehension and recall," *Arab World English Journal*, vol. 9, pp. 234–250, 06 2018.

[22] F. P. Anderson, R. C., "Vocabulary knowledge," *In J. T. Guthrie (Ed.), Comprehension and Teaching: Research Reviews (pp. 77-117). Newark, DE: International Reading Association*, 1981.

[23] X. Lu, "The relationship of lexical richness to the quality of esl learners' oral narratives," *The Modern Language Journal*, vol. 96, no. 2, pp. 190–208, 2012. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-4781.2011.01232_1.x

[24] E. Gibson, "Linguistic complexity: Locality of syntactic dependencies," *Cognition*, vol. 68, pp. 1–76, 09 1998.

[25] D. BIBER, B. Gray, and K. Poonpon, "Should we use characteristics of conversation to measure grammatical complexity in l2 writing development?" *TESOL Quarterly*, vol. 45, 03 2011.

[26] R. Kate, X. Luo, S. Patwardhan, M. Franz, R. Florian, R. Mooney, S. Roukos, and C. Welty, "Learning to predict readability using diverse linguistic features," in *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*. Beijing, China: Coling 2010 Organizing Committee, Aug. 2010, pp. 546–554. [Online]. Available: https://www.aclweb.org/anthology/C10-1062

[27] X. Lu, "Automatic analysis of syntactic complexity in second language writing," *International Journal of Corpus Linguistics*, vol. 15, no. 4, pp. 474–496, 2010. [Online]. Available: https://www.jbe-platform.com/content/journals/10.1075/ijcl.15.4.02lu

[28] I. S. . K. H.-Y. Wolfe-Quintero, K., *Second language development in writing: Measures of fluency, accuracy, & complexity*. Honolulu: Second Language Teaching & Curriculum Center, University of Hawaii at Manoa, 1998.

[29] L. Ortega, "Syntactic Complexity Measures and their Relationship to L2 Proficiency: A Research Synthesis of College-level L2 Writing," *Applied Linguistics*, vol. 24, no. 4, pp. 492–518, 12 2003. [Online]. Available: https://doi.org/10.1093/applin/24.4.492

[30] F. Shadloo, H. Shahriari, and B. Ghonsooly, "Exploring syntactic complexity and its relationship with writing quality in efl argumentative essays," *Topics in Linguistics*, vol. 20, pp. 68–81, 06 2019.

[31] F. Kuiken, I. Vedder, A. Housen, and B. Clercq, "Variation in syntactic complexity: Introduction," *International Journal of Applied Linguistics*, 04 2019.

[32] A. C. Graesser, D. S. McNamara, M. M. Louwerse, and Z. Cai, "Coh-metrix: Analysis of text on cohesion and language," *Behavior Research Methods, Instruments, and Computers*, vol. 36, pp. 193–202, 2004.

[33] D. McNamara, M. Louwerse, P. McCarthy, and A. Graesser, "Coh-metrix: Capturing linguistic features of cohesion," *Discourse Processes*, vol. 47, no. 4, pp. 292–330, 5 2010.

[34] B. Klebanov and E. Shamir, "Reader-based exploration of lexical cohesion," *Language Resources and Evaluation*, vol. 41, pp. 27–44, 10 2007.

[35] R. Barzilay and M. Lapata, "Modeling local coherence: An entity-based approach." *Computational Linguistics*, vol. 34, pp. 1–34, 01 2008.

[36] B. J. Grosz, A. K. Joshi, and S. Weinstein, "Centering: A framework for modeling the local coherence of discourse," *Computational Linguistics*, vol. 21, no. 2, pp. 203–225, 1995. [Online]. Available: https://www.aclweb.org/anthology/J95-2003

[37] M. Elsner and E. Charniak, "Extending the entity grid with entity-specific features." in *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*, 01 2011, pp. 125–129.

[38] D. Nguyen and S. Joty, "A neural local coherence model," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017, pp. 1320–1330. [Online]. Available: http://www.aclweb.org/anthology/P17-1121

[39] M. Mesgar and M. Strube, "A neural local coherence model for text quality assessment," in *Proceedings of the 2018 Conference on Empirical Methods in*

Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 4328–4339.

[40] E. Pitler and A. Nenkova, "Revisiting readability: A unified framework for predicting text quality," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. Honolulu, Hawaii: Association for Computational Linguistics, Oct. 2008, pp. 186–195.

[41] L. Feng, M. Jansche, M. Huenerfauth, and N. Elhadad, "A comparison of features for automatic readability assessment," in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, ser. COLING '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 276–284. [Online]. Available: http://dl.acm.org/citation.cfm?id=1944566.1944598

[42] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, vol. 5, pp. 157–66, 02 1994.

[43] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[44] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. [Online]. Available: https://www.aclweb.org/anthology/D14-1179

[45] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 2048–2057.

[46] M. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *CoRR*, vol. abs/1508.04025, 2015. [Online]. Available: http://arxiv.org/abs/1508.04025

[47] M. Martinc, S. Pollak, and M. Robnik-Sikonja, "Supervised and unsupervised neural approaches to text readability," *CoRR*, vol. abs/1907.11779, 2019. [Online]. Available: http://arxiv.org/abs/1907.11779

[48] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 09 2017, pp. 670–680.

[49] C. o. C. S. S. O. National Governors Association Center for Best Practices, *Common Core State Standards Appendix B.* National Governors Association Center for Best Practices, Council of Chief State School Officers, Washington D.C., 2010.

[50] C.-Y. Lin and E. Hovy, "The automated acquisition of topic signatures for text summarization," in *COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics*, 2000. [Online]. Available: https://www.aclweb.org/anthology/C00-1072

[51] C. A. Engber, "The relationship of lexical proficiency to the quality of esl compositions," *Journal of Second Language Writing*, vol. 4, no. 2, pp. 139 – 155, 1995. [Online]. Available: http://www.sciencedirect.com/science/article/pii/1060374395900047

[52] M. Linnarud, *Lexis in composition : a performance analysis of Swedish learners' written English.* Lund : Gleerup, 1986.

[53] K. Hyltenstam, "Lexical characteristics of near-native second-language learners of swedish," *Journal of Multilingual and Multicultural Development*, vol. 9, no. 1-2, pp. 67–84, 1988. [Online]. Available: https://doi.org/10.1080/01434632.1988.9994320

[54] B. Laufer, "The lexical profile of second language writing: Does it change over time?" *RELC Journal*, vol. 25, no. 2, pp. 21–33, 1994. [Online]. Available: https://doi.org/10.1177/003368829402500202

[55] B. Harley and M. L. King, "Verb lexis in the written compositions of young l2 learners," *Studies in Second Language Acquisition*, vol. 11, no. 4, p. 415–439, 1989.

[56] C. Chaudron and K. Parker, "Discourse markedness and structural markedness: The acquisition of english noun phrases," *Studies in Second Language Acquisition*, vol. 12, no. 1, p. 43–64, 1990.

[57] T. Klee, "Developmental and diagnostic characteristics of quantitative measures of children's language production." *Topics in Language Disorders*, vol. 12, no. 12, p. 28–41, 1992.

[58] J. F. Miller, "Quantifying productive language disorders." *In J. F. Miller (Ed.), Research in child language disorders: A decade of progress*, p. 211–220, 1991.

[59] D. Malvern, B. Richards, N. Chipere, and P. Duran, *Lexical diversity and language development: Quantification and assessment.* Basingstoke, Hampshire: Palgrave Macmillan, 05 2004.

[60] E. Thordardottir and S. Ellis Weismer, "High-frequency verbs and verb diversity in the spontaneous speech of school-age children with specific language impairment," *International journal of language & communication disorders / Royal College of Speech & Language Therapists*, vol. 36, pp. 221–44, 04 2001.

[61] J. B. .-. Carroll, *Language and thought.* Englewood Cliffs, N.J. : Prentice-Hall, 1964, bibliography: p. 112-113. [Online]. Available: http://digitool.hbz-nrw.de:1801/webclient/DeliveryManager?pid=2540963&custom_att_2=simple_viewer

[62] G. Herdan, *Quantitative linguistics.* London: Butterworths., 1964.

[63] W. Johnson, "Studies in language behavior: I. a program of research," *Psychological Monographs*, vol. 56, no. 2, pp. 1–15, 1944.

[64] P. Guiraud, *Problèmes et méthodes de la statistique linguistique [Problems and methods of statistical linguistics].* Dordrecht, The Netherlands: D. Reidel., 1960.

[65] M. C. Templin, *Certain Language Skills in Children: Their Development and Interrelationships.* University of Minnesota Press, 1957, vol. 26. [Online]. Available: http://www.jstor.org/stable/10.5749/j.ctttv2st

[66] D. Dugast, *Vocabulaire et stylistique. I Théâtre et dialogue [Vocabulary and style. Vol. 1 Theatre and dialogue].* Geneva, Switzerland: Slatkine-Champion., 1979.

[67] E. McClure, "A comparison of lexical strategies in l1 and l2 written english narratives." *Pragmatics and Language Learning*, vol. 2, no. 2, p. 141–154, 1991.

[68] C. P. Casanave, "Language development in students' journals," *Journal of Second Language Writing*, vol. 3, no. 3, pp. 179 – 201, 1994. [Online]. Available: http://www.sciencedirect.com/science/article/pii/1060374394900167

[69] M. Hubert and E. Vandervieren, "An adjusted boxplot for skewed distributions," *Computational Statistics & Data Analysis*, vol. 52, no. 12, pp. 5186 – 5201, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167947307004434

[70] G. Brys, M. Hubert, and A. Struyf, "A robust measure of skewness," *Journal of Computational and Graphical Statistics*, vol. 13, no. 4, pp. 996–1017, 2004.

[71] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960. [Online]. Available: https://doi.org/10.1177/001316446002000104

[72] R. Artstein and M. Poesio, "Inter-coder agreement for computational linguistics," *Comput. Linguist.*, vol. 34, no. 4, p. 555–596, Dec. 2008. [Online]. Available: https://doi.org/10.1162/coli.07-034-R2