Array 12 (2021) 100097

Contents lists available at ScienceDirect

Array

journal homepage: www.elsevier.com/locate/array

DeepEva: A deep neural network architecture for assessing sentence complexity in Italian and English languages

Giosué Lo Bosco [a,d], Giovanni Pilato [b,*], Daniele Schicchi [c,d]

[a] Dipartimento di Matematica e Informatica, Universitá degli studi di Palermo, Via Archirafi 34, 90123 Palermo, Italy
[b] CNR, Istituto di Calcolo e reti ad Alte Prestazioni, Consiglio Nazionale delle Ricerche, Via Ugo La Malfa 153, 90146 Palermo, Italy
[c] CNR, Istituto di Tecnologie Didattiche, Via Ugo la Malfa 153, 90146, Palermo, Italy
[d] IEMEST, Istituto Euro Mediterraneo di Scienza e Tecnologia, Via Michele Miraglia 20, 90139, Palermo, Italy

ARTICLE INFO

Keywords:
Text-complexity-assessment
Automatic-text-complexity-evaluation
Text-simplification
Artificial-intelligence
Deep-learning
Natural-language-processing

ABSTRACT

Automatic Text Complexity Evaluation (ATE) is a research field that aims at creating new methodologies to make autonomous the process of the text complexity evaluation, that is the study of the text-linguistic features (e.g., lexical, syntactical, morphological) to measure the grade of comprehensibility of a text. ATE can affect positively several different contexts such as Finance, Health, and Education. Moreover, it can support the research on Automatic Text Simplification (ATS), a research area that deals with the study of new methods for transforming a text by changing its lexicon and structure to meet specific reader needs. In this paper, we illustrate an ATE approach named DeepEva, a Deep Learning based system capable of classifying both Italian and English sentences on the basis of their complexity. The system exploits the Treetagger annotation tool, two Long Short Term Memory (LSTM) neural unit layers, and a fully connected one. The last layer outputs the probability of a sentence belonging to the *easy* or *complex* class. The experimental results show the effectiveness of the approach for both languages, compared with several baselines such as Support Vector Machine, Gradient Boosting, and Random Forest.

1. Introduction

Recent years have been characterized by the significant growth of solutions related to Natural Language Processing (NLP) problems. Such solutions vary and they concern different topics such us computational creativity [1], support system for teaching [2–4], machine translation [5], semantic analysis [6], health support system [7] and so on. Automatic Text Simplification (ATS) is a natural language processing task whose main purpose is to transform a text in an automated manner to make it more easily understandable by a reader, keeping as much as possible the original meaning of the content. People with language disabilities, low reader skills, or lack of knowledge of a specific language are categories of users who can benefit from ATS systems.

Nowadays, researches related to the ATS concern the development of intelligent systems capable of simplifying texts automatically. In this context, the automatic text complexity evaluation (ATE) is a relevant research field related to the development of Text Simplification systems. ATE systems analyze the features of the text that are representative of its complexity and relate them with those linked to the reading skills of the user. An ATE recognizes if a text is already suitable for the reader or it needs to be simplified. If the text is judged to be too difficult, the text should be modified, for instance, by changing the lexicon or the sentence's syntax to adapt the text complexity to the reading skills of the user.

An ATE system is not only related to the ATS activities: it can be used for many different contexts as a standalone system. The automatic evaluation of the complexity of a text can be appreciated as *support system* by people that have to engage with different communities. Educators are an example of individuals who need ATE systems since they often produce educational material that can be used by students having linguistic problems such as those affected by dyslexia, deafness, or aphasia. In this regard, ATE supports educators during the drafting process by suggesting simplifying the text if it is not suitable for a reader's skills. Furthermore, substantial waves of immigration have occurred in Italy since 2017.[1] These phenomena have increased the number of students, who are not native speaker inside classrooms, that have to tackle linguistic understanding problems. An augmentation of students in need of supporting demands more effort to educators to

* Corresponding author.
E-mail addresses: giosue.lobosco@unipa.it (G. Lo Bosco), giovanni.pilato@icar.cnr.it (G. Pilato), daniele.schicchi@itd.cnr.it (D. Schicchi).

[1] http://www.libertaciviliimmigrazione.dlci.interno.gov.it/it/documentazione/statistica/cruscotto-statistico-giornaliero

https://doi.org/10.1016/j.array.2021.100097
Received 1 August 2021; Accepted 28 September 2021
Available online 12 November 2021

prepare the educational material, which increases the worth of ATE tools.

Although investments have improved the school system making education available to almost everybody, there is still a high percentage of the population with low reading skills. Statistical investigation[2] has been carried out to assess literacy competencies of 24 OCSE countries. The study places Italy as the country with the highest number of people with worse literacy skills. At the same time, England/Ireland and the United States are ranked respectively 15th and 17th for language competencies showing the need for simplification tools for both Italian and English languages.

In this paper, we propose a solution for overcoming the classical measures to assess text complexity leveraging Deep Learning and an effective parsing tool named Treetagger [8]. Facets that make a sentence not suitable for the reader are identified via a learning process that exploit a dataset which include the description of the reader skills through the examples. The system considers the syntactical features by using Treetagger that extracts the sentence's parts-of-speech. Instead, it exploits an RNN to extract the most important facets of text complexity useful to classify sentences as *hard to understand* (i.e, the sentence does not meet the reader skills) or *easy to understand* (i.e. the sentence is suitable for the reader) for both languages. The architecture of the network uses two Long Short Term Memory (LSTM) neural unit layers and a fully connected one. The LSTM layers analyze lexical and syntactic peculiarities by exploiting its capabilities of remembering the input sequence arrangement. The output, representing the extracted features, stimulates the next layer, activated using the softmax activation function, which gives the probability of a sentence belonging to one of the two classes.

Experimental results highlight the system aptitude for the ATE task. Such a system achieves relevant values of the F1-Score measure for both English and Italian, underlining its versatility for tackling the problem in more than one language.

The paper is structured as follows: in Section 2 we bibliography contents of ATE, Sections 3.1 and 3.2 describe the system with a focus on the NN architecture, Section 4 gives information about the used corpus for training the NN and the experiments carried out for evaluating its performance; a discussion on the effectiveness of the NN is provided in Section 5. Finally the conclusions are given in Section 6.

2. Literature review

2.1. Historical measures

ATE is a relevant research topic that has been studied for the English language since 1893 [9]. The first important attempt to tackle the problem was leveraged on quantitative approaches that exploited statistical formulas to measure the text readability mainly for the English language. Subsequently, researchers started to study ATE for other languages such as Italian.

In 1943, Rudolf Flesch created a readability formula which take into account three language elements: *average sentence length in words*, *number of affixes*, and *number of references to people* [10]. Such a formula became state of the art. It was utilized to control the text complexity in many different contexts like newspaper reports, government publications, and materials for adult education. In the same line, in 1969, Bormuth [11] created a statistical formula which exploits *average sentence length*, *number of words* not on the revised Dale list of words known by fourth graders, and *number of letters per word*.

In 1975, by taking inspiration from the original Flesch formula, it was created the *Flesch–Kincaid readability index* (FKI) [12], one of the most common indexes for assessing text complexity. It is based on the combination of three common readability formulas: the *Automated Readability Index, Fog Count* and *Flesch Reading Ease*. It was developed in the military environment to help the training of Navy enlisted personnel. The problem was to create a reliable support tool for making training material understandable by the military students. FKI gives a score that measures the complexity level of an English document based on its structural features. It takes into account *number of words*, *number of sentences*, *amount of syllables*, and relates them by using numerical coefficients calculated carrying out tests on 531 military personnel. Despite the relevance of this index, it has some limitations. One is that it evaluates longer texts as more challenging to understand. Note that the length of a text is not always a feature that affects the text complexity. The text might be longer because of a larger amount of information, which could help the reader better understand. Furthermore, this typology of indexes does not take into account other important facets of text complexity. Indeed, it is a common opinion that evaluating only document surface features is not enough for assessing its complexity. Despite its limitations, the FKI has been largely adopted.

In 1986, Roberto Vacca adapted the FKI for the Italian language creating the *Flesch–Vacca index* (FVI) [13], which assesses specifically the text complexity of an Italian text. Another historical index created for measuring the complexity of an Italian text is *GulpEase* (GE) [14]. It is based on the *length of words*, the *number of words* and the *number of sentences*. GE was created specifically for the Italian language, and it provides a score that can be used to associate the complexity of text with the degree level of the reader. For example, a score of 60/100 points attests that the text complexity is very difficult for people with an elementary school diploma but easy to understand for people with a high school diploma. Unfortunately, also GulpEase lacks in considering other fundamental aspects of text complexity.

In 1989, the Lexile framework, a formula which use *sentence length* and *word frequency*, was developed [15]. The formula uses 1000 Lexile points (from 200 to 1200), where 200 is the first grade, and 1200 is the twelfth-grade level. The Lexile score has been used for assessing 30'000 books, and a growing number of test publishers such as CTB-McGraw Hill and NWEA Achievement Level Test have adopted such a measure.

In 1994, a formula based on the Bormuth formula, named Degrees of Reading Power, was created to assess the text complexity by exploiting *sentence length*, *number of words* not in an updated version of the Dale List, and *average number of letters per word* [16]. This formula uses a point-scale of difficulty on the range 15–85, and the authors provide a translation table which maps the formula values to the reading levels.

The most important shortcomings of the described formulas are related to the use of only surface facets like the *sentence length* and *number of syllables*. These are not enough to cover all the factors that characterize the text complexity. Therefore they have become outdated, and researchers have started to explore new methodologies to assess the text complexity.

2.2. Modern measures

The shortcomings of traditional readability indices are overcome by using cognitive studies on how the reader interacts with a text. Such studies conclude that the assessment of text comprehension can not use only shallows features but they must include psycholinguistics ones, such as *text cohesion*, *syntactic parsing*, *measures related to decoding*, and *meaning construction*.

In [17], the *CohMetrix* is utilized to measure English text readability for students who are learning a second language (L2 readers) relying on text cohesion. It integrates many deep-level factors like *semantic lexicons*, *part-of-speech taggers*, and other computational linguistics components that allow the examination of features related to text processing and reading comprehension.

Meaningful readability index has been developed according to Prototype Theory [18], which states that the most readable words represent objects humans interact with. For example, *guitar* is more readable

[2] https://www.oecd.org/skills/piaac/Country%20note%20-%20Italy%20(ITA).pdf

than its superordinate words *stringed instrument* and its subordinate words *acoustic guitar*. Then, Wordnet [19] is exploited to study such words relations in order to build a readability measure which relates *basic*, *superordinate* and *subordinate* words.

Statistical techniques have been explored for creating the language model of a particular grade level. The analysis of large corpora suitable for a specific type of reader allows us to discover features that characterize his abilities. In [20], the reading difficulty of Web pages has been assessed by using simple statistical language models and surface linguistic features. An extension of Bayes classification has been used for combining multiple language models to determine the text complexity [21]. In [22] Grammar and vocabulary features are combined for estimating the reading difficulty to outperform models based only on grammar or language modeling approaches. In [23], the problem of readability is modeled as a text-categorization task tackles by using a statistical language model based on a variation of the multinomial naïve Bayes classifier. Such a model has been utilized to classify Web pages by examining their reading difficulty. In [24], lexical (i.e., the relative frequencies of word unigrams) and grammatical features (i.e., extracted from automatic context-free grammar parses of sentences) are correlated utilizing statistical models to build a measure of readability. The study compares Linear Regression, Multiclass Logistic Regression, and Proportional Odds to choose the model the best suit the problem. Statistical models have been widely utilized for evaluating the readability of financial disclosure since the way they are written, and content comprehension affects the decision to invest in a product [25,26].

The multiple types and nature of features that could be involved in text complexity evaluation lead to the suggestion that a machine learning model might catch the relation among features by its learning process. Following this suggestion, several supervised models have been provided. In [27], Simple English Wikipedia and Wikipedia are used to create a corpus that contains sentences labeled respectively as simple and hard, then lexical, syntactic, and psycholinguistic features are extracted to train the SVM model for classifying sentences based on their difficulty. In [28] a dependency tree and a semantic network are used to build a readability index in which features like *sentence length*, *word length* and *word frequency* are related by using the nearest neighbor algorithm. The study shows that deep syntactic and semantic features help better represent a reader's difficulties in understanding a text. In [29], a Stochastic Gradient Descent classifier is proposed for the binary classification of complex and simple sentences. The training of the algorithm has been carried out on a specific corpus created aligning sentences of Newsela [30] through massAlign [31] system. In [32], features like word length, sentence length, part-of-speech counts, frequency of common words, medical concept density, specificity, and ambiguity are used to train six different ML algorithms for predicting the difficulty of health texts. In [33], a new readability assessment approach that relies on a set of features to support the process of text simplification with cognitively-motivated metrics has been used for supporting the text-simplification process for poor literacy readers. The assessment has been carried out through a standard classifier, an ordinal (ranking) classifier, and a regressor. The best model is then embedded in an efficient Text Simplification system.

A robust system developed for measuring sentence complexity of the Italian language is READ-IT [34]. READ-IT is an SVM-based system capable of taking into account many text features that affect sentence complexity. The training phase has been carried out by exploiting the *Repubblica* newspaper, which is considered difficult to comprehend for the 70% of the Italian people, and *Due Parole* which articles are recommended to low literacy skills readers. Starting from an Italian sentence READ-IT extracts *raw*, *lexical*, *morpho-syntactic*, and *syntactic* features which are used for training the SVM to identify *hard to understand* and *easy to understand* sentences. It offers a score of complexity, represented by the probability of the sentence being one of the two classes. In the context of the evaluation of the readability of software codes, in [35] many classifiers are tested to discover the features of code writing, which affect the software quality and to create a readability measure. The study presents a descriptive model of software readability, which is strongly correlated to the judgments of 120 human annotators.

The sentence complexity evaluation has been explored by authors interpreting it as a classification problem. The aim is to classify sentences in two classes based on their *lexical* and *syntactical* features. We have chosen NN models since they have been successfully used as supervised classification models showing their power in many contexts. Furthermore, they make it possible to overcome the weakness of the features extraction phase since the model itself automatically accomplishes it. In [36] *Lexical* and *syntactical* features have been evaluated, analyzing the sentence as a sequence of tokens, where a token is either a word or a punctuation symbol. The system is characterized by a preprocessing phase that represents a sentence as a sequence of real number vectors and by a Recurrent Neural Network (RNN), which analyzes the sequence to learn what is the peculiarities that make the sentence *hard* or *easy* to understand. According to the computed tests, the system performances are coherent with the READ-IT ones, a state-of-the-art system for measuring the text complexity for the Italian language, establishing it as a good alternative to measure the sentence complexity level for the Italian language.

A system based on NN architecture has been developed for measuring *syntactic* complexity [37,38]. These kinds of systems could be used to support authors who create texts for people with problems to comprehend syntactic constructs. The system works well for both Italian and English languages, showing the NN models' high versatility to tackle the problem for different languages. The developed model has been compared to the SVM baseline system, which achieves as good as NN model performances for the English language but not for the Italian one.

To improve the NN model performances, authors have investigated how evaluating the text complexity is related to the tokens representation [39]. The paper describes a set of experiments in which the evaluation of the text complexity has been carried out through the same model but with different tokens representation methodologies. The results suggest that the problem of text complexity evaluation is mainly affected by the model architecture than the ways to represent the sentence elements.

Nowadays, tools for understanding if the text generated by an ATS system is effectively suitable for the target reader are necessary to enhance the TS research field. In addition to the system already described, there exist other metrics explicitly created for this purpose, like *FKBLEU* and *SARI* [40]. FKBLEU attempts to combine the Flesch–Kincaid index with an extension of the well-known BLEU index to create a new measure capable of capturing the *readability* and *adequacy* of the simplified text. The SARI index takes into account how good is the system in *adding*, *deleting* or *keeping* words that support the simplification, and it evaluates the system employing a rewards-based methodology.

3. Method

We present *DeepEVA*, a text complexity evaluation system that classifies sentences based on their difficulty. Two main modules compose the system: the preprocessing and the classification model. The preprocessing module enriches the sentences with its *parts-of-speech*, and it deals with the adaptation of sentences to make them suitable for the analysis by the classifier. The final output of the preprocessing is a representation of data in a vectorial form. The second module is a supervised classifier based on an RNN, which learns how to discriminate *hard to understand* sentences from the *easy to understand* ones by examining labeled sentences. The structure of our system is shown in Fig. 1.

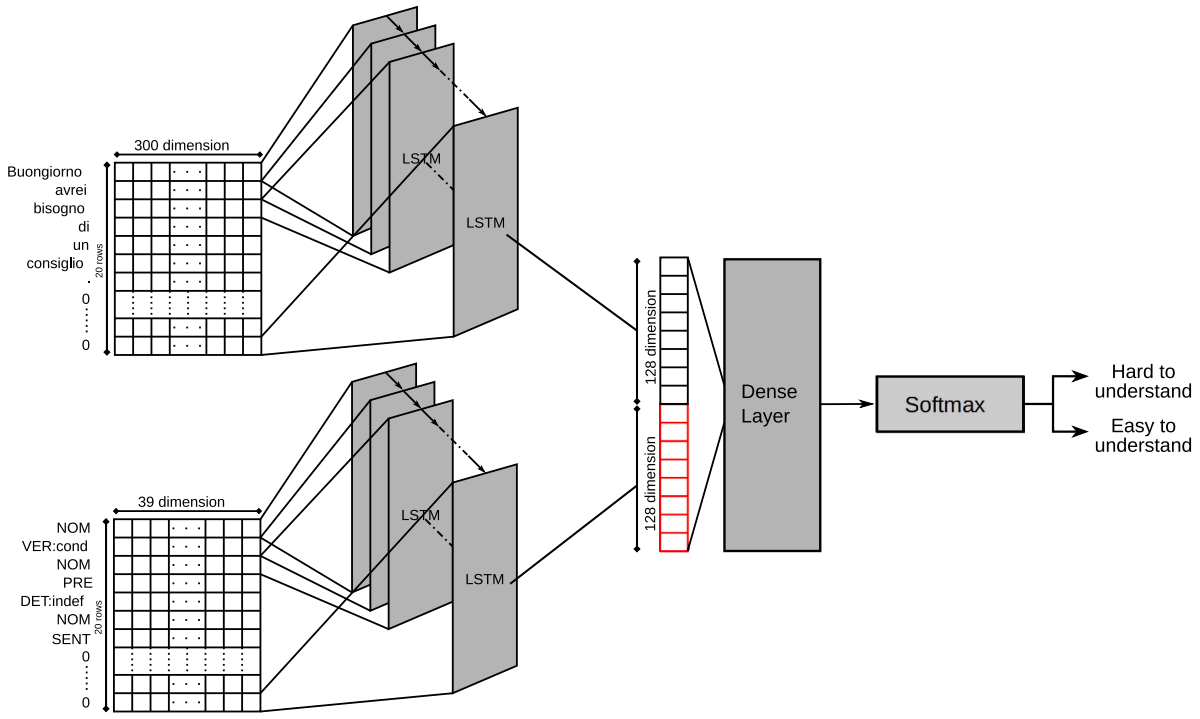

Fig. 1. The system architecture. The picture is divided into modules. From left to right: the representation matrix of a sentence computed by the preprocessing module by transforming words, punctuation marks, and part-of-speech into a vector form. The Model module describes the neural networks which learn what is *hard to understand* and *easy to understand*. The output module evaluates the response of the Model assigning the sentence to either the first or the second class.

3.1. Preprocessing

The preprocessing phase is done before the model starts the elaboration of the sentence. Its objectives are multiple: to carry out a deep analysis of the sentence extrapolating its *parts-of-speech*, to recognize *words* and *punctuation* marks, to transform each *part-of-speech*, *word* and *punctuation* symbol to an appropriate vector of numbers that can be analyzed by the model and which properly represent the features of the input sentence.

The process of extrapolating the sentence *parts-of-speech* and recognize *words* and *punctuation* marks is done by using a pre-trained version of Treetagger [8]. Treetagger is an annotating tool that has been created to associate *parts-of-speech* to sentence tokens. In our case, each token is either word or a punctuation mark of the sentence. The choice of this tool is due to its ability to tag different languages like German, English, Italian, and so on. Since the idea of the authors was to create a unique model capable of understanding what are the features that identify a sentence as *hard to understand* or *easy to understand* for Italian and English language, the preprocessing phase needs to be done for both the languages in a coherent way.

Treetagger is highly customizable, which means that the tool can be used for different languages, simply changing a configuration file, that we call *tag-set*, which describes the features of the language. Each *tag-set* file is explicitly created for a language, and it identifies a set of linguistic elements that can be recognized during the analysis of the text.

The Italian language has been tagged using the Stein,[3] *tag-set* this configuration file takes into account linguistic elements such as *adverbs, adjective, verb* and *noun*. In detail, Tretagger using the Stein configuration file can recognize 13 different categories of *verbs*, 8 different types of *pronouns*, *numeral*, *punctuation*, *name*, definite and indefinite *article*, *abbreviation*, *adjective* and so on. In addition to Stein configuration file, there exist another *tag-set* file created by Baroni[4] that we have not used for our experiments. The reason justifying this choice can be found in [37] where the authors show that the Stein configuration file is more suitable for the problem we are tackling.

The same procedure has been done for the English language. We have used Treetagger for tagging the English sentences that will be elaborated by the *model*. The configuration file utilized for this process is the one trained on the British National Corpus named *BNC tag-set*.[5] The *tag-set* is composed by 61 tag including, inter alia, 25 different categories of *verbs*, *adverb*, *noun*, 4 different *punctuation* marks types, *prepositions* and 4 classes of *nouns*.

After the tagging process, every *part-of-speech* associated with a sentence is coded as a vector utilizing the well known *one-hot encoding*. This type of coding system consists of creating a *vector* with a total length equal to the amount of *parts-of-speech* recognizable by the tool for a specific language. The rationale of the methodology is to consider the vector positions representing all possible *parts-of-speech* and the value of 1 as a mark point, which suggests the presence of a determined *part-of-speech*. Thus, the vector elements are put to 0 except for a *unique* position that contains the value of 1.

Words and *punctuation* marks have been also detected by using Treetagger. Both of them are turned into vectors of real numbers by using FastText [41]. FastText is a library that allows making effective word representations and sentence classification, taking into account features not only related to the entire *word* or *punctuation* marks but also tied to local characteristics like the bag of characters that compose the *word*. It has been used for representing *words* and *punctuation* marks of 157 different languages [42] such as Italian and English. This work has produced available resources in which, for a specific language, there is a correspondence between a word and a vector of real numbers. In detail, FastText [41,42] has been trained on Wikipedia,[6]

[3] https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/data/italian-tagset.txt

[4] http://sslmit.unibo.it/~baroni/collocazioni/itwac.tagset.txt

[5] http://www.natcorp.ox.ac.uk/docs/gramtag.html

[6] https://www.wikipedia.org/

and Common Crawl[7] and it maps each token of the sentence to a 300-dimensional space vector which means that, in our case, at the end of the preprocessing phase the meaning and the structure of the sentence will be represented as a sequence of 300-dimensional vector suited for the analysis by the model.

3.2. Classifier

The classifier module is based on a specific type of NN known as *Long–Short Term (LSTM) Neural Networks* [43]. *LSTM* Neural Networks belong to the family of Recurrent Neural Networks (RNNs), a set of networks that tackle the problem of analyzing sequences. Their peculiarity is the exploiting of a stimulus called *feedback* that constitute a cumulative representation of the sequence elements that the network has already analyzed. This type of recurrence allows creating a link between the outcome of the network and each element of the input sequence, which means a different result if the elements or their order change. The computation of the RNNs can be understood by using the *unfolding* concept [44] in which the network is evaluated as a progression of states which was taken during the analysis of the sequence elements. RNNs can consider all the elements of the sequence, changing their behavior based on the sequence elements and their order. Remembering how the elements are arranged into the sequence shows extraordinary potential to model problems related to speech recognition [45], language model generation [46], machine translation [47], emotion recognition in a video [48] and so on.

Despite their good features, these types of networks are difficult to train using back-propagation through time (BPTT) [49] because of the well-known *vanishing gradient* problem [50]. There exists an optimization algorithm (e.g., [51]) that avoid the problem of *vanishing gradient*. However, the high effectiveness of these algorithms is comparable to their computational cost, so they are less attractive than the BPTT methodology. To leave the methodology unchanged, the researchers have designed a new architecture of RNN units called *LSTM cell* that is capable of facing the *vanishing gradient* problem holding the main properties related to the recurrence.

The *LSTM cell* goes beyond the *vanish gradient* problem by means of specific architecture based on *gates*. The system of gates controls how the information is propagated from input to the output and acts on the internal state of the *cell*. As shown in Fig. 2, the *LSTM cell* contains two loops (*o-loop* and *s-loop*) which allow to implement the *feedback stimulus* to keep track of the sequence of elements. The *input* is related to the *gates* by a series of operations which affect the state of the system and the output of the *cell* is mainly given by an appropriate combination of the *output gate* and the system state. More details can be found in [44].

The classifier model we propose will be able to differentiate between two classes of sentences: *easy to understand*, *hard to understand*. As explained in Section 2, the complexity of a sentence is influenced by many factors such as the *lexicon* and its *syntax*, therefore it is important to consider methodologies that can include not only words but also the structure of the sentence. The RNN has shown good potential to understand these features, as shown in [36–38] in which the authors present that a NN architecture is reliable to classify sentences based on their *difficulty*. The proposed classifier model tries to combine the ideas of [36–38] to build a new powerful model that can outperform the performance of past models.

For this reason, we propose a classifier that is composed by two separate LSTM layers, L_1 and L_2. L_1 layer deals with the examination of *parts-of-speech* sequence while the layer L_2 analyzes the progression of both *words and punctuation symbols*

The model learns separately the features that represent the difficulty level of comprehension behind the *parts-of-speech* and the complexity

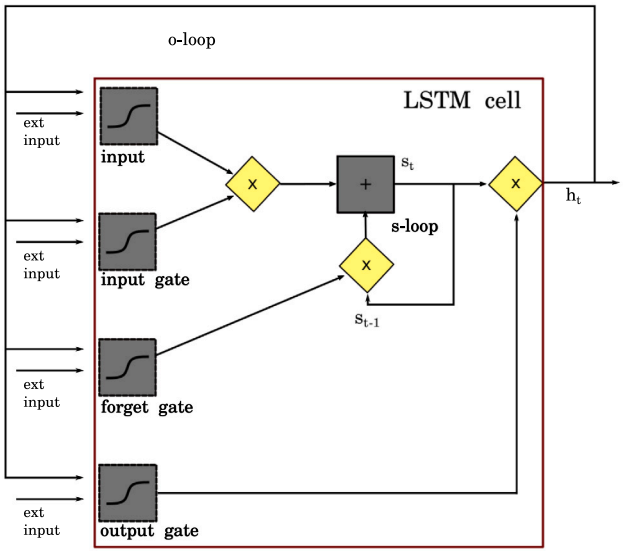

Fig. 2. The LSTM Cell. The picture shows the flow of the operations that are applied to the input sequence. Each input element contribute to the final outcome and it is related to the *input*, *forget* and *output* gate which are activated by a sigmoid function and the *input* which is activated by a logistic sigmoid function.

aspects expressed by *words and punctuation marks*. Analyzing the *parts-of-speech* the L_1 layer discovers the syntactic rules which makes a sentence more or less *hard to understand* while the layer L_2 looking at *words and punctuation marks* finds out features related to both *lexical* and *syntactic* aspects that identify the complexity of the sentence. Indeed, the syntax of a sentence is related to its structure; thus, the layer L_2 observing how the tokens follow one another can make inference on the sentence syntax. The outcomes of the layers L_1 and L_2 are concatenated, and then they are processed by the successive dense layer, which is fully connected with the previous. The Dense Layer takes care of evaluating the contributes of L_1 and L_2 layers so that mix the information in order to give a judgment about the complexity of the sentence. The output of this layer is activated by the *softmax* function, which gives the probability that a sentence belongs to the category of *hard to understand* or *easy to understand*. The Output module deals with the evaluation of the dense layer output. Its objective is to compare the two outputs of the dense layer and to choose the maximum. This means assigning the input sentence to the class that is more likely to be correct based on the knowledge that the network has acquired.

3.3. Parameters

The system parameters have been computed empirically by testing different loss functions, optimization algorithms and trying multiple combinations of neuron numbers. Tests have shown that an efficient solution is to use 128 neurons for each layer and training the network, minimizing the well-known *cross-entropy* loss function choosing the RMSPROP [52] optimization algorithm on balanced minibatch of size 50. In doing so, the model architecture relies on the first two layers composed of 128-LSTM neural units whose outcome is analyzed by the last 2-units dense layer activated by the softmax function and normalized by using an L2 norm with a factor scale of 0.05. The Early-stopping approach has been used, setting the threshold at 0.001 to avoid overfitting.

We have considered the maximum length of each sentence the average length of the entire corpus, which means 20 tokens for the English language and 21 for the Italian one.

[7] www.commoncrawl.org

4. Results

4.1. Corpus

In the text complexity evaluation, the concepts of *hard* and *easy to understand* are strongly related to the reader's linguistic skills. Specific types of texts might be challenging for a class of people but very understandable for others characterized by higher linguistic proficiency. The same scenario is reflected for the ATE systems, which have to consider the target people's properties (e.g., L1 learners, L2 learners, dyslexic and deaf people) to compute the complexity threshold that, if exceeded, identifies text as not suitable for the reader. Such readability skills could be embedded directly within the system or described in a concealed manner inside the dataset using the labeling process. An ATE system based on ML algorithms learns from data on classifying sentences based on their complexity. It sets the most appropriate threshold by examining the labels associated with the data. The computed threshold is representative of the knowledge acquired by the model, and it may be explained only if the labeling process is described accurately. For these reasons, both training and test phases have been carried out by using two accurately chosen corpora: a specific Italian corpus annotated according to the Common European Framework of Reference for Languages (CEFR) which contains only Italian sentences and the English corpus that is composed based on Newsela, which content has been accurately drafted by a group of professional linguistics.

4.1.1. Italian

Unfortunately, the evaluation of text complexity is made harder by the lack of resources exploitable for the Italian language. To the best of our knowledge, the only corpus big enough for training a deep learning model is PACCSS [53] that has been created in a semi-automatic way for solving specifically the ATS task. PACCSS has also been used in the context of ATE [36,37,39] by using as *hard to understand* all the not-simplified sentences and as *easy to understand* the relative simplified versions. Although it represents a resource to solve the problem, the need for a specific Italian corpus for ATE persists, since PACCSS is mainly adequate for the ATS topic and it is a silver-standard.

We have created a new sentence-based corpus by harvesting publicly online resources to develop a more reliable model. The corpus is a mixture of texts drafted specifically for teaching the Italian language, fairy tales for children, and classical Italian novels. While the teaching material is handmade annotated, we have manually examined the reminder documents tagging the fables for children as *A2* and the classic Italian novels as *C2*. It is well known that fables addressing low-proficiency linguistic users are written by using straightforward syntactical constructs and common words. Whereas we have selected, classical Italian novels considered *complicated* by the majority of Italians (e.g., Anna Karenina).

The corpus is enriched by the sentences extracted from *dueparole*[8](2P). 2P is a news magazine whose articles are written using a clear, precise, and *easy to understand* language. 2P aims to make accessible the information to people who have difficulties in comprehending Italian texts. For example, they could be people who are not mother tongues, people with language disabilities such us *dyslexia* or *aphasia* and mother tongue people who have low linguistic skills. The authors of 2P are professional linguists, journalists, and teachers. Their studies have led to discovering a set of criteria for controlling the complexity of texts and communicating effectively. They use a specific method of writing, keeping the text short, with *easy to understand* sentences, and they enrich the text with common Italian words, which are more easily understandable. The method, called *controlled writing*, can be applied to different types of text like informative, regulatory, bureaucratic, and so on.

[8] www.dueparole.it

The final corpus contains about 100.000 sentences whose lengths are comprised between 6 and 177 tokens and distributed as follows: C1/C2: 73.000, B2/B1: 1.000, A2/A1:26.000. The total amount of different words is around 70.000, and it concerns both L1 and L2 learners. The new corpus, annotated, according to the CEFR standard, represents a reliable resource for training and testing machine learning models suitable for the subject problem.

4.1.2. English

For what concerns the English language, we have trained the model using the *Newsela* [30] corpus. The idea behind the creation of the corpus is to help educators for preparing students to meet the English language objectives for each grade level according to Common Core Standards [54] in the United States. Since the reliability and quality of the resource, it has become widely used for helping the TS field. Indeed, before the Newsela corpus, the most important resource suitable to resolve TS problems was the *Parallel Wikipedia Simplification* (PWS) corpus created aligning articles from Wikipedia and Simple Wikipedia.[9] PWS is built using automatic sentence alignment methodologies, which make it prone to contain errors. Indeed, many simplifications are inappropriate experiencing the 50% of the sentence pairs are not simplification [30].

The Newsela authors overcome these troubles by creating a simplification corpus with the aid of professional editors without the use of automatic methodologies. The corpus is designed for children at different grade levels. It is composed of 1.130 news articles which each of which has been rewritten at least four times[10] by professionals at different grades of complexity meeting the needs of different reader levels. The documents are labeled with numbers ranging from 0 to 4, denoting the complexity of the text. 0 represents the document's original version, while the labels from 1 to 4 mean successive simplification levels of the same document where 4 (or 5 in some cases) is the easiest version of the document.

Unfortunately, the Newsela corpus is a set of articles that is not compatible with the idea of the authors that is to create a model capable of evaluating the complexity of sentences. Indeed, the Newsela corpus specifies the complexity of the entire document, not giving any information about the complexity of the sentences included in the article. This means that although an article is classified as a specific complexity level L, its sentences do not necessarily reflect the same complexity the set of all sentences reflect L. To tackle this problem, we have processed Newsela articles extracting the sentences and then trying to divide them into the classes *hard to understand* and *easy to understand*.

The sentence extraction has been carried out using a regular expression that selects all the sentences of documents as a sequence of characters terminated by a dot mark, treating apart special cases like acronyms. After the process, we harvested approximately 530,000 sentences associate with the complexity level of the document, where the sentence is extracted.

The dataset consists in a list of pairs (s_i, d_{jk}) formed by the sentences s_i, that are included in the document d_{jk} that has a complexity level $k \in \{0, 1, 2, 3, 4, 5\}$.

We have analyzed the sentences inside the documents looking for how they are being distributed. Table 1 shows the number of common sentences between all possible complexity pairs. In particular, the element (L_x, L_y), $x, y \in 0, 1, 2, 3, 4, 5$ represent the number of common sentences between all the documents with complexity level x and all documents with complexity level y. For example, the element table (L_2, L_3) shows that the number of common sentences among the documents with complexity level 2 and the documents with complexity level 3 is 22869.

[9] https://simple.wikipedia.org/wiki/Main_Page

[10] There are exceptions in which the document has five degrees of simplification.

Table 1
Every table cell identified by the row L_x and column L_y, represents the number of the same sentences between all the document of difficulty level x and those of difficulty level y.

#	L_0	L_1	L_2	L_3	L_4	L_5
L_0	104801	32475	17991	8493	4191	41
L_1	32475	99980	31849	11918	5151	43
L_2	17991	31849	108586	22869	8274	58
L_3	8493	11918	22869	111029	21253	97
L_4	4191	5151	8274	21253	103496	254
L_5	41	43	58	97	254	2073

Table 2
The averaged measures on English (Left) and Italian (Right) corpora computed by means of 10-fold cross validation methodology which shows the variation of F1-SCORE on changing of the neurons number and training epochs. The results are achieved by means of Early Stopping process.

Epochs	Neurons	F1-SCORE	Epochs	Neurons	F1-SCORE
15	16	.880	20	16	.865
11	32	.880	13	32	.864
9	64	.879	11	64	.864
7	128	.880	10	128	.866
4	256	.879	7	256	.866

The analysis of the distribution shows, as expected, the higher number of common sentences for couples (x, x), symmetric values between couples (x, y) and (y, x), $x, y = 1, 2, 3, 4, 5$, and a low number of common sentences between couples (x, y), $x \geq 4$ and $y \leq 2$.

In order to reach our purposes we categorized as *hard to understand* all the sentences present in document d_{jk}, with grade of difficulty $k \leq 1$ that are not included in documents d_{jk} with $k \geq 2$. More formally, we set the set of hard to understand sentences set H as:

$$H = \bigcup_{\forall j,k \leq 1} d_{jk} \setminus \bigcup_{\forall j,k \geq 2} d_{jk}$$

We consider *easy to understand* all the sentences included in documents d_{jk}, that have a grade of difficulty $k \geq 4$ which are not inside documents d_{jk}, with $k \leq 3$, i.e.

$$E = \bigcup_{\forall j,k \geq 4} d_{jk} \setminus \bigcup_{\forall j,k \leq 3} d_{jk}$$

By using the above-described selection paradigm, the cardinality of H is 130.000, while E is composed of 80.000 sentences. The final corpus contains sentences of lengths between 1 and over 160 tokens; the vocabulary size is equal to 92817 different words.

4.2. Experiments

The objective of the paper is to build a model capable of tackling the problem of evaluating the complexity of sentences in Italian and English languages. In the following, we present experiments and results.

As described in Section 3.1, parameters of the model have been chosen empirically. We have carried out a set of experiments to evaluate the system's performances by changing the number of LSTM neural units. Table 2 reports for each model configuration both the performances and the epochs needed to attain the best system. Based on the achieved results, we have chosen to set the neurons number of the LSTM layers at 128, limiting both the complexity of the network and the computational effort for the training process. Indeed the system achieves the best performances after seven epochs for the English language and ten epochs for the Italian language. Every experiment takes into account the set of *hard to understand* sentences as *positive* class and the set of *easy to understand* sentences as *negative* class.

Table 3
Results achieved by both DeepEva and the baseline model on Italian harvested Corpus. Every measure is carried out as the average of results computed for each run on the base of the 10-Fold method.

Model	Epochs	Kernel	Recall	Precision	F1-SCORE
DeepEva-IT	10	–	.872 (±.014)	.862 (±.009)	.862 (±.004)
SVM-IT-L	–	linear	.725 (±.003)	.789 (±.002)	.756 (±.002)
SVM-IT-R	–	rbf	.708 (±.002)	.624 (±.003)	.663 (±.002)
RF-IT	–	–	.756 (±.003)	.762 (±.003)	.759 (±.003)
GB	–	–	.794 (±.003)	.753 (±.002)	.773 (±.002)

4.2.1. Experiment 1: Italian

The corpus contains Italian sentences classified following the six[11] ascending levels of difficulty described on the CEFR standard. In order to create the binary classification model, we have selected as E=*easy to understand* (Negative class) the sentences labeled as the *A1*, *A2* and *B1*, while we have chosen as H=*hard to understand* (Positive class) the ones classified as *B2*, *C1* or *C2*. The derived corpus includes around 26.500 *easy to understand* sentences and 73.000 *hard to understand*.

Since the dataset is unbalanced, in order to exploit the entire dataset, we have partitioned the biggest class H into L sub-sets S_i, where $|S_i|$ is equal to $|E|$, and carried out several *runs*. Each run exploits a dataset composed of a partition S_i and the entire set of the *easy to understand* sentences. As *runs* change, a new dataset is created by coupling an S_i that has not been chosen before and E. For each created dataset, we considered using the K fold cross-validation approach averaging the partial results. Final results are computed as the averaged partial results over the number of runs. The quantification of the performance is done using Accuracy, Recall, Precision. The Precision and the Recall are used to calculate the F1-SCORE.

For what concerns the Italian language, we have used a 10-Fold for each of the three *runs*.

DeepEva has been trained for a variable number of epochs. For each epoch, according to the K-Fold methodology, we have trained K models.

The system has been compared with a baseline method that relies on SVM, Random Forest, and Gradient Boosting. Each input sentence is represented numerically by using a pre-trained embedding tool computed using FastText [42] and the bag of words method. Every sentence is embedded as the concatenation of a) the normalized sum of all the vectors representing the tokens and b) the bag of words applied to the parts of speech normalized as well. The baseline method has been tested on more runs as well as the DeepEva system to exploit the entire Italian corpora. To improve the performances of the SVM baseline, the following kernel methods have been used: *linear*, and *rbf*. For what concerns the Random Forest and Gradient boosting both exploit 100 estimators. Their implementation has been carried out by exploiting scikit-learn[12] [55] and holding the standard parameters.

Table 3 contains results achieved by both DeepEva and the baseline models on the Italian corpus, showing the better performances achieved by the NN-based model.

DeepEva has been tested exploiting the PACCSS corpus as well. As described above, the achieved performances are compared to baseline models ones that rely on SVM (with linear and RBF kernel), Random Forest, and Gradient Boosting. They all exploit the same sentence representation manner, which relies on FastText and the bag of words of parts of speech. In this case, DeepEva achieves the best results after 19 epochs.

[11] CEFR levels: A1, A2, B1, B2, C1, C2
[12] scikit-learn version 0.23.2

Table 4
Results achieved by DeepEva compared to the ones achieved by the baseline on PACCSS. Every measure is carried out as the average of results computed on the base of 10-fold method.

Model	Epochs	Kernel	Recall	Precision	F1-SCORE
DeepEva-PIT	19	–	.880 (±.012)	.895 (±.011)	.888 (±.002)
SVM-IT-PL	–	linear	.725 (±.002)	.789 (±.002)	.756 (±.002)
SVM-IT-PR	–	rbf	.708 (±.004)	.624 (±.003)	.663 (±.002)
RF-IT-P	–	–	.857 (±.001)	.877 (±.002)	.867 (±.000)
GB-IT-P	–	–	.741 (±.004)	.823 (±.005)	.780 (±.003)

Table 5
Results achieved by both DeepEva and the baseline model on the English Corpus. Every measure is carried out as the average of results computed for each run on the base of 10-Fold method.

Model	Epochs	Kernel	Recall	Precision	F1-SCORE
DeepEva-EN	7	–	.870 (±0.010)	.890 (±0.008)	.880 (±0.002)
SVM-EN-L	–	linear	.797 (±0.002)	.892 (±0.002)	.842 (±0.001)
SVM-EN-R	–	rbf	.789 (±0.001)	.806 (±0.001)	.797 (±0.001)
RF-EN	–	–	.794 (±0.02)	.869 (±0.001)	.823 (±0.001)
GB-EN	–	–	.803 (±0.002)	.867 (±0.001)	.834 (±0.001)

4.2.2. Experiment 2: English

The performance evaluation of the system for the English language has been carried out on sentences inside the Newsela corpus. The test-set is created using the extraction procedure described in Section 4.1. According to that used for the Italian language, the system testing has been carried out by using the K-Fold cross-validation methodology with $K = 10$. Since the number of *hard to understand* sentences is greater than the *easy to understand* one the experiments has been carried out in two *runs* (see Section 4.2.1). The first run exploits a dataset composed of all the T *easy to understand* sentences and the first T *hard to understand* sentences. In the second run, the model uses a dataset composed by taking the latest T *hard to understand* sentences and the set of the *easy to understand* sentences. Finally, the results have been averaged on K-Fold and then over the number of runs. The quantification of the performance is computed using Recall and Precision. The last ones are used to calculate the F1-Score.

The performances of DeepEva are compared with the same baseline models used for experiments in the Italian language. The baseline models have been trained on two runs and carried out previously to evaluate the entire set of available data. Table 5 shows both the most relevant result achieved by DeepEva and the comparison with the baseline models.

5. Discussion

The first insight that comes from experiments is the suggestion that different efforts need to evaluate sentence complexity written in Italian and English. The above tables show that the highest performances are achieved faster for the English language than for the Italian one that needs more training epochs. Moreover, let us look at the F1-Scores. The model finds it harder to infer the features that denote the complexity of the Italian sentences since it achieves worse classification measures making more effort.

The system has been tested to measure its effectiveness for the evaluation task. To increase the robustness of the testing process, the entire set of experiments exploit the cross-validation methodology with a considerable number of Folds. Moreover, since the datasets are unbalanced, the K-Fold method has been applied on more *runs* of the model execution to use all the available data (see Section 4.2.1).

In the first experiment (Section 4.2.1), the system has been evaluated by measuring its performance to assess the complexity of sentences written in the Italian language. Results in Table 3 show the best value of F1-Score for the harvested corpus is achieved after training the model for ten epochs, Table 4 contains the best system's performance achieved for the PACCSS corpus by training the network for 19 epochs. Such high values of F1-Score suggest the system's capabilities of evaluating the complex facets of the Italian sentences inferred from the data. The Italian corpora are composed of many different types of sentences, which include several perspectives of both *lexical* and *syntactical* complexity. Moreover, the system addresses a crucial feature of the topic; it is not *length-dependent* since for both the *hard to understand* and the *easy to understand* classes, the number of tokens ranges widely. Indeed, although longer sentences might be believed *harder to understand*, they could contain a significant amount of information that limits the growth of the complexity.

The model has been compared with baseline models trained by following the methodology used for the performance evaluation of DeepEva and the same amount of information (i.e., embedding and parts of speech). Experiments have taken into account Random Forest, Gradient Boosting, and SVM with different kernels methods. Notice that DeepEva has been trained and tested from scratch for both corpora because the harvested corpus and PACCSS address different types of readers.

Results show that the best performances have been achieved by Gradient Boosting for the harvest corpus and by Random Forest for the PACCSS one. Nonetheless, the measures are not as good as the ones achieved by DeepEva that can evaluate the sentence complexity better than other models. Only the Random Forest trained on PACCSS achieves results comparable to the ones of DeepEva. This is interesting and worth a deeper analysis that will be carried out in future works.

The second experiment aids the evaluation of the performances of DeepEva for the English language. Experiments have been carried out by following the methodology established for the first experiment. Therefore, a 10-Fold cross-validation methodology has been used repeatedly to take into account the entire dataset (see Section 4.2.2). Table 5 shows the overall performances of DeepEva for the evaluation of the English sentence complexity. Training the network for seven epochs allows us to attain an F1-Score value equal to 0.88. It can be deducted the capabilities of the network to recognize the features that affect the text complexity. Specifically, there are few discrepancies between the Recall and the Precision measures, which means that the system can recognize *hard to understand* sentences keeping a high level of exactness. Considering both the methodology to carry out the experiment and the reliability of the data exploited for the training, such high values of the achieved measures guarantee an accurate functioning of the system. Baseline models are used to evaluate the effectiveness of DeepEva. In contrast to results shown for the Italian language, the SVM-L can reach higher results for the English language. Table 5 outlines the best F1-Score if the baseline is used in combination with the linear kernel. Overall, Random Forest and Gradient Boosting achieve almost the same value of F1-Score of SVM-L, showing good behavior for classifying sentences. In this case, the Precision value is comparable to the one achieved by DeepEva; on the contrary, the Recall measure is 8% worse.

6. Conclusions

We have introduced an ATE system based on RNN capable of taking into account many facets of complexity to classify sentences based on their difficulties. The evaluation of sentence complexity is a new perspective of the ATE research fields, which offers different challenges than those based on documents. Working on sentences is more complex since there is less information for the analysis, but it is more beneficial since the system allows us to intervene directly on the part of the text that needs to be simplified. The system has been tested for Italian and English languages, the high performances of the system to classify *hard* and *easy to understand* sentences suggest its high versatility to tackle the problem of complexity evaluation for different languages. Although the languages are very different from various points of view, the system

shows the ability to discover the features that affect complexity for both of them.

Future works will be focused on carrying out additional tests to understand the system's functioning deeply. We are going to compare DeepEVA with state-of-the-art AI methods such as transformers and convolutional neural networks for text classification. A deeper linguistic analysis will be carried out on the employed corpora. Moreover, the system is being embedded in a more complex system whose aim is to simplify text in an automated manner under development, and it will be also used in conversational systems like those used in [56,57] in order to improve the effectiveness of interaction with users in social robotic environments [58].

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.array.2021.100097.

Acknowledgments

This work has been developed in the framework of the project COURAGE - A social media companion safeguarding and educating students (no. 95567), funded by the Volkswagen Foundation in the topic Artificial Intelligence and the Society of the Future. This research is also funded in part by MIUR Project of National Relevance 2017WR7SHH "Multicriteria Data Structures and Algorithms: from compressed to learned indexes, and beyond". We also acknowledge an NVIDIA Higher Education and Research Grant (donation of a Titan V GPU).

References

[1] Schicchi D, Pilato G. WORDY: A semi-automatic methodology aimed at the creation of neologisms based on a semantic network and blending devices. In: Barolli L, Terzo O, editors. Complex, intelligent, and software intensive systems. Cham: Springer International Publishing; 2018, p. 236–48.
[2] Schicchi D, Pilato G. A social humanoid robot as a playfellow for vocabulary enhancement. In: 2018 second IEEE International conference on robotic computing. Los Alamitos, CA, USA: IEEE Computer Society; 2018, p. 205–8.
[3] Schicchi D, Pilato G. Portmanteau word-play for vocabulary enhancement with humanoid robot support. Encycl Semant Comput Robotic Intell 2018;2(01):1850006.
[4] Alonso JM, Casalino G. Explainable artificial intelligence for human-centric data analysis in virtual learning environments. In: Burgos D, Cimitile M, Ducange P, Pecori R, Picerno P, Raviolo P, Stracke CM, editors. Higher education learning methodologies and technologies online. Cham: Springer International Publishing; 2019, p. 125–38.
[5] Di Gangi MA, Federico M. Deep neural machine translation with weakly-recurrent units. In: 21st annual conference of the european association for machine translation. 2018. p. 119–28.
[6] Di Gangi MA, Lo Bosco G, Pilato G. Effectiveness of data-driven induction of semantic spaces and traditional classifiers for sarcasm detection. Nat Lang Eng 2019;25(2):257–85.
[7] Alfano M, Lenzitti B, Lo Bosco G, Perticone V. An automatic system for helping health consumers to understand medical texts. In: HEALTHINF 2015 - 8th international conference on health informatics, proceedings; Part of 8th international joint conference on biomedical engineering systems and technologies. 2015. p. 622–7.
[8] Schmid H. Probabilistic part-of-speech tagging using decision trees. In: New Methods in Language Processing. 2013, p. 154.
[9] Sherman LA. Analytics of literature: A manual for the objective study of English prose and poetry. Ginn; 1893.
[10] Flesch R. Marks of readable style; a study in adult education. In: Teachers college contributions to education. 1943.
[11] Bormuth JR. Development of readability analyses. Final Report, Project No. 7-0052, Contract (1), 1969.
[12] Kincaid J. Derivation of new readability formulas: (Automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Research Branch report, Chief of Naval Technical Training, Naval Air Station Memphis; 1975.
[13] Franchina V, Vacca R. Adaptation of Flesh readability index on a bilingual text written by the same author both in Italian and English languages. Linguaggi 1986;3:47–9.
[14] Lucisano P, Piemontese ME. GULPEASE: una formula per la predizione della difficoltà dei testi in lingua italiana. Scuola e città 1988;3(31):110–24.
[15] Smith DR, et al. The lexile scale in theory and practice. Final report, ERIC; 1989.
[16] TASAssociates. DRP handbook. Touchstone Applied Science Associates; 1994.
[17] Crossley SA, Greenfield J, McNamara DS. Assessing text readability using cognitively based indices. TESOL Q 2008;42(3):475–93.
[18] Lin S-Y, Su C-C, Lai Y-D, Yang L-C, Hsieh S-K. Assessing text readability using hierarchical lexical relations retrieved from WordNet. Int J Comput Linguist Chin Lang Proc 2009;14(1):45–84.
[19] Miller GA. WordNet: A lexical database for English. Communications of the ACM 1995;38(11):39–41.
[20] Si L, Callan J. A statistical model for scientific readability. In: Proceedings of the tenth international conference on information and knowledge management. New York, NY, USA: ACM; 2001, p. 574–6.
[21] Collins-Thompson K, Callan JP. 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; 2004, p. 193–200.
[22] Heilman M, Collins-Thompson K, Callan J, Eskenazi M. Combining lexical and grammatical features to improve readability measures for first and second language texts. In: Human language technologies 2007: The conference of the North American chapter of the association for computational linguistics; proceedings of the main conference. Rochester, New York: Association for Computational Linguistics; 2007, p. 460–7.
[23] Collins-Thompson K, Callan J. Predicting reading difficulty with statistical language models. Journal of the American Society for Information Science and Technology 2005;56(13):1448–62.
[24] Heilman M, Collins-Thompson K, Eskenazi M. 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. 2008. p. 71–9.
[25] Bonsall IV SB, Leone AJ, Miller BP, Rennekamp K. A plain English measure of financial reporting readability. J Account Econ 2017;63(2–3):329–57.
[26] Loughran T, McDonald B. Measuring readability in financial disclosures. J Finance 2014;69(4):1643–71.
[27] Vajjala S, Meurers D. Assessing the relative reading level of sentence pairs for text simplification. In: Proceedings of the 14th conference of the European chapter of the association for computational linguistics. 2014. p. 288–97.
[28] Vor der Brück T, Hartrumpf S, Helbig H. A readability checker with supervised learning using deep indicators. Informatica 2008;32(4).
[29] Scarton C, Paetzold G, Specia L. Text simplification from professionally produced corpora. In: Proceedings of the eleventh international conference on language resources and evaluation. 2018.
[30] Xu W, Callison-Burch C, Napoles C. Problems in current text simplification research: new data can help. Transactions of the Association for Computational Linguistics 2015;3:283–97.
[31] Paetzold G, Alva-Manchego F, Specia L. MASSAlign: Alignment and annotation of comparable documents. In: Proceedings of the IJCNLP 2017, system demonstrations. Tapei, Taiwan: Association for Computational Linguistics; 2017, p. 1–4.
[32] Kauchak D, Mouradi O, Pentoney C, Leroy G. Text simplification tools: Using machine learning to discover features that identify difficult text. In: 2014 47th Hawaii international conference on system sciences. IEEE; 2014, p. 2616–25.
[33] Aluisio S, Specia L, Gasperin C, Scarton C. Readability assessment for text simplification. In: Proceedings of the NAACL HLT 2010 fifth workshop on innovative use of NLP for building educational applications. 2010. p. 1–9.
[34] Dell'Orletta F, Montemagni S, Venturi G. Read-it: Assessing readability of italian texts with a view to text simplification. In: Proceedings of the second workshop on speech and language processing for assistive technologies. Association for Computational Linguistics; 2011, p. 73–83.
[35] Buse RP, Weimer WR. Learning a metric for code readability. IEEE Transactions on Software Engineering 2009;36(4):546–58.
[36] Lo Bosco G, Pilato G, Schicchi D. A recurrent deep neural network model to measure sentence complexity for the italian language. AIC 2018, Artificial Intelligence and Cognition 2018 - CEUR Workshop Proceedings 2018;2418:90–7.
[37] Lo Bosco G, Pilato G, Schicchi D. A sentence based system for measuring syntax complexity using a recurrent deep neural network. In: 2nd workshop on natural language for artificial intelligence, vol. 2244. CEUR-WS; 2018, p. 95–101.
[38] Schicchi D, Lo Bosco G, Pilato G. Machine learning models for measuring syntax complexity of English text. In: Samsonovich AV, editor. Biologically inspired cognitive architectures 2019. Cham: Springer International Publishing; 2020, p. 449–54.
[39] Lo Bosco G, Pilato G, Schicchi D. A neural network model for the evaluation of text complexity in Italian language: a representation point of view. In: Postproceedings of the 9th Annual international conference on biologically inspired cognitive architectures, BICA 2018 (Ninth Annual Meeting of the BICA Society). Procedia Comput Sci 2018;145:464–70.
[40] Xu W, Napoles C, Pavlick E, Chen Q, Callison-Burch C. Optimizing statistical machine translation for text simplification. Transactions of the Association for Computational Linguistics 2016;4:401–15.

[41] Bojanowski P, Grave E, Joulin A, Mikolov T. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics 2017;5:135–46.
[42] Grave E, Bojanowski P, Gupta P, Joulin A, Mikolov T. Learning word vectors for 157 languages. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). Miyazaki, Japan: European Language Resources Association (ELRA); 2018, https://aclanthology.org/L18-1550.
[43] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9(8):1735–80.
[44] Goodfellow I, Bengio Y, Courville A. Deep Learning. MIT Press; 2016.
[45] Graves A, Mohamed A, Hinton G. Speech recognition with deep recurrent neural networks. In: 2013 IEEE international conference on acoustics, speech and signal processing. 2013, p. 6645–9. http://dx.doi.org/10.1109/ICASSP.2013.6638947.
[46] Kipyatkova I, Karpov A. Recurrent neural network-based language modeling for an automatic Russian speech recognition system. In: 2015 artificial intelligence and natural language and information extraction, social media and web search FRUCT conference. 2015, p. 33–8. http://dx.doi.org/10.1109/AINL-ISMW-FRUCT.2015.7382966.
[47] Cho K, van Merrienboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: Proceedings of the 2014 conference on empirical methods in natural language processing. Doha, Qatar: Association for Computational Linguistics; 2014, p. 1724–34. http://dx.doi.org/10.3115/v1/D14-1179.
[48] Ebrahimi Kahou S, Michalski V, Konda K, Memisevic R, Pal C. Recurrent neural networks for emotion recognition in video. In: Proceedings of the 2015 ACM on international conference on multimodal interaction. New York, NY, USA: ACM; 2015, p. 467–74. http://dx.doi.org/10.1145/2818346.2830596.
[49] Rumelhart DE, Hinton GE, Williams RJ. In: Anderson JA, Rosenfeld E, editors. Neurocomputing: Foundations of research. Cambridge, MA, USA: MIT Press; 1988, p. 696–9.
[50] Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw 1994;5(2):157–66. http://dx.doi.org/10.1109/72.279181.
[51] Martens J, Sutskever I. Learning recurrent neural networks with Hessian-free optimization. In: Proceedings of the 28th international conference on international conference on machine learning. USA: Omni Press; 2011, p. 1033–40.
[52] Hinton G, Srivastava N, Swersky K. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. Coursera Lecture slides; 2012.
[53] Brunato D, Cimino A, Dell'Orletta F, Venturi G. PaCCSS-IT: A parallel corpus of complex-simple sentences for automatic text simplification. In: Proceedings of the 2016 conference on empirical methods in natural language processing. Association for Computational Linguistics; 2016, p. 351–61.
[54] Porter A, McMaken J, Hwang J, Yang R. Common core standards: The new U.S. intended curriculum. Educ Res 2011;40(3):103–16.
[55] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E. Scikit-learn: Machine learning in Python. J Mach Learn Res 2011;12:2825–30.
[56] Pilato G, Pirrone R, Rizzo R. A kst-based system for student tutoring. Applied Artificial Intelligence 2008;22(4):283–308.
[57] Vassallo G, Pilato G, Augello A, Gaglio S. Phase coherence in conceptual spaces for conversational agents. Semantic computing 2010;357–71.
[58] Cuzzocrea A, Pilato G. A composite framework for supporting user emotion detection based on intelligent taxonomy handling. Logic J IGPL 2021;29(2):207–19.