ISSN 2007-9737

How Much Deep is Deep Enough?

Diego Uribe, Enrique Cuan

TecNM Instituto Tecnológico de La Laguna,
Mexico

{duribea, ecuand}@lalaguna.tecnm.mx

Abstract. Typical deep learning models defined in terms of multiple layers are based on the assumption that a better representation is obtained with a hierarchical model rather than with a shallow one. Nevertheless, increasing the depth of the model by increasing the number of layers can lead to the model being lost or stuck during the optimization process.This paper investigates the impact of linguistic complexity characteristics from text on a deep learning model defined in terms of a stacked architecture. As the optimal number of stacked recurrent neural layers is specific to each application, we examine the optimal number of stacked recurrent layers corresponding to each linguistic characteristic. Last but not least, we also analyze the computational cost demanded by increasing the depth of a stacked recurrent architecture implemented for a linguistic characteristic.

Keywords. Recurrent neural networks, stacked architectures, linguistic characteristics.

1 Introduction

Nowadays, the successful application of deep learning models performing tasks without human intervention is part of our daily lives. For example, the use of deep learning models such as convolutional networks in visual recognition exhibited a spectacular success in the largest contest in object recognition known as ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [16]. Natural Language Processing is another difficult task in which the implementation of deep learning models such as recurrent neural networks (RNN) has contributed to improve the state of the art in multiple challenges of natural language understanding.

For example, the work of Jain et al. to produce short stories from short remarks is an evidence of how text generation has exhibited great progress [10]. The work of Wang et al. to predict sentiment polarity from tweets is also an illustration of how sequence classification tasks have attested good results [23].

Bengio, one of the pioneers in the field, argues the necessity of deep architectures for more efficient representation of AI-high-level tasks by making use of multiple hidden layers instead of shallow models [1, 2]. In our case, we study the use of stacked recurrent architectures, that is, deep architectures based on stacking multiple recurrent hidden layers on top of each other.

Since recurrent neural networks, ss fundamental deep learning algorithms for sequence processing, can be built in many different ways, we analyze three basic and popular recurrent architectures: simple recurrent networks [5], Long Short-Term Memory (LSTM) networks [9], and Gate Recurrent Unit (GRU) networks [3]. LSTM y GRU are more sophisticated architectures that have been created to cope not only with the necessity of memory “to remember” previous elements in a sequence but also with the “vanishing gradient problem” that is experimented with neural networks of a big depth, i.e. those feedforward networks with many hidden layers.

To determine the right depth of a stacked recurrent architecture to text processing is the main focus in this investigation. Specifically, our purpose is to provide empirical evidence about the relationship between the linguistic characteristics of the text and the stacked recurrent learning models.

In other words, the research question addressed in this paper is: what is the proper depth of a stacked recurrent architecture corresponding to a particular level of linguistic complexity inherent in the texts?

Nonetheless, in order to give answer to this question, it is first important to elucidate another concern: how to define the linguistic complexity of the textual expressions? Based on the literature review, we have identified some linguistic properties to determine the complexity occurring in the texts.

How much hard is the comprehension of a text? Text comprehension is a crucial linguistic property to determine the complexity of a text. Paul Rhea argues that the difference between easy and complex sentences is related to the number of embeddings, that is, the number of nested clauses within the sentence [19]. For example, the difference between the following sentences is notorious: the second sentence is more complex and demands more computational processing.

(i) *The sad reality is that most Canadians simply don' t care about access to information.*

(ii) *The sad reality is that most Canadians, as long as they can watch Don Cherry on Sat nite and as long as Tim Hortons keeps their donut prices affordable, they simply don' t care about access to information.*

Is it necessary to increase the depth of a recurrent learning model when we process texts similar to the first sentence? About texts containing long-term dependencies as the second sentence, will it be necessary to increase the number of stacked recurrent layers?

In short, which stacked recurrent architectures coping with particular linguistic characteristics provide more representational power than a single-layer recurrent model? To give answer to these questions, it is paramount to determine the linguistic properties of the texts to be processed.

In addition to the assessment of complexity based on the number of embedded clauses in the texts (as we explained above), we also explore the entropy of the texts as a metric of how rich the vocabulary of texts is.

Taking as reference the work of Keller [12] in which he makes evident a correlation between the entropy of a sentence and the complexity required for its comprehension, the consideration of entropy in this work is based on the assumption that if the text is more complex, the author uses a more varied vocabulary.

This study also consider the use of quantitative methods as assessment of textual complexity such as the analysis of how long a text is. In fact, the number of sentences and the number of content words in the text are regarded as textual properties to determine the impact on the performance of a stacked recurrent architecture.

And last but not least, we also analyze the computational cost demanded by increasing the depth of a stacked recurrent architecture put into practice for each linguistic characteristic. In summary, the main contributions of this research work are the following:

— Providing empirical evidence about the impact of linguistic complexity characteristics from texts on a deep learning model defined in terms of a stacked recurrent architecture.

— Using qualitative (embedded clauses and entropy) and quantitative (number of sentences and words) methods, multiple linguistic complexity characteristics of the texts are considered in this research.

— Applying deep learning algorithms (RNN, LSTM and GRU), various stacked recurrent architectures are implemented in this study.

2 Related Work

In this section we first proceed with a brief description of preceding works about the investigation of deep networks for machine learning purpose. Then we comment on previous works about text analysis and the use of linguistic properties for the computational processing of the texts.

2.1 Stacked Architectures

Utgoff and Stracuzzi presented many-layered learning as the need of many layers of knowledge for learning non-trivial concepts [22]. When a difficult problem is broken into a sequence of simple problems, learning is modeled with layered knowledge structures defined in terms of interdependent and reusable concepts named building blocks. In this way, assimilation of new knowledge makes use of previous knowledge.

By using equivalent Boolean functions, where one of them is defined in terms of nested basic elements in a more compact expression, a model learning for the assimilation of the target concept require different number of layers for each equivalent function. In fact, a nested Boolean function, that denotes a complex concept described in terms of simple elements (i.e. building blocks), requires more learning layers to assimilate and reuse the target concept sometime thereafter.

In contrast, the equivalent Boolean function, that has not been described in terms of building blocks, requires less learning layers (i.e. shallow network) to assimilate a concept hardly reusable. In other words, since knowledge reuse is an essential factor for achieving successful learning, this investigation shows how shallow networks make difficult the learning process.

Graves et al. showed how a deep learning model based on a stacked recurrent architecture was effective for phoneme recognition [8]. Even tough RNNs are a type of neural networks suitable for sequential data, in speech recognition better results were obtained by deep feedforward networks (vanilla neural networks). This antecedent was the main reason to investigate the use of deep recurrent neural networks for speech recognition. In particular, a deep LSTM architecture was applied to speech recognition and better performance was obtained over learning models based on single-layer LSTM.

Two key elements were considered in the definition of a deep learning model for speech recognition. First, with the use of a recurrent architecture as LSTM, not only the analysis of previous elements is considered but also the analysis of a long range context in the sequence. Furthermore, the LSTM architecture was enriched with bidirectional layers so the elements in the sequence were examined in both directions.

Secondly, a deep architecture was defined by stacking multiple bidirectional recurrent layers on top of each other, with the output sequence of one layer forming the input sequence for the next. By stacking recurrent layers, the model generates multiple levels of representation which proved to be relevant in the processing of the phonemes. In this way, the combination of multiple levels of representation with long range context analysis of the acoustic sequence was essential for building up an effective stacked recurrent architecture for speech recognition.

Pascanu et al. explored different ways to extend a recurrent neural network (RNN) to a deep RNN [18]. This research was inspired by previous works showing how increasing the depth of a classic neural network proved to be more efficient at representing some functions than a shallow one. Based on the processing of a simple RNN, they analyzed the basic steps carried out by an RNN in order to identify points of deeper extensions.

Since the basic steps: input-to-hidden function, hidden-to- hidden transition and hidden-to-output function are all shallow, that is, there is no intermediate layer, an alternative deeper design was proposed for each shallow point, and in this way, deeper variants of an RNN.

For example, the hidden-to-hidden transition was made deeper by having one or more intermediate nonlinear layers between two consecutive hidden states (h_{t-1} and h_t).

Two deeper variants of an RNN were empirically evaluated on the tasks of polyphonic music prediction and language modeling. The experimental results proved how the depth of the proposed variants of an RNN was essential to outperform the shallow RNNs. Another interesting outcome of the experimentation was that each of the proposed deep RNNs has a distinct characteristic that makes it more, or less, suitable for certain types of datasets.

2.2 Linguistic Features

We now briefly describe some interesting works about text complexity and constructiveness classification. Santucci et al. investigated the complexity level of an Italian text from the classification task perspective [20]. The experimentation made use of a collection of texts produced for second language teaching purpose and a large set of linguistic features were defined to be used among ten machine learning models where the random forest stood out from the rest. But beyond the good accuracy results obtained, the key point was the conduction of a deep analysis to identify the set of linguistic features that influenced the good prediction results.

Another interesting investigation was carried out by Yasseri et al. They investigated the complexity of two categories of English texts from Wikipedia: Simple and Main texts where both text samples exhibit rich vocabulary. Statistical analysis of linguistic units such as n–grams of words and part of speech tags provided empirical evidence of how the language of Simple texts is less complex due to the use of shorter sentences.

With the comparison of these categories by the Gunning readability index was evident the linguistic complexity not only in terms of the linguistic units but also in terms of the topic addressed in texts: the language of conceptual articles is more elaborate compared to biographical and object–based articles [24].

Kolhatkar and Taboada implemented classical (SVM classifiers) and deep (biLSTMs) learning models to recognize constructive comments by making use of two datasets for training: the New York Times Picks as positive examples and the Yahoo News Annotated Comments Corpus as negative examples of constructive online comments.

The learning models were evaluated on a crowd-annotated corpus containing 1,121 comments. In this investigation, multiple sets of constructiveness features were defined: length, argumentation, named–entity and text–quality features. The purpose of such sets was the identification of those crucial features to determine how argumentative or constructive a comment is [13]. These sets of constructiveness features provided the inspiration for a deep work.

3 Learning Models and Linguistic Characteristics

This section describes the stacked recurrent architectures and the linguistic characteristics of the texts contemplated in this study. As we previously mentioned, the motivation of this research is to investigate the impact of linguistic complexity characteristics from texts on a deep learning model defined in terms of a stacked recurrent architecture. We first explain each recurrent neural network and the corresponding linguistic peculiarities that caused it. Then, we describe the linguistic characteristics for the analysis of how complex a text is.

3.1 Stacked Recurrent Architectures

Stacked Recurrent Architectures are the focus of our attention in this work. A model based on vanilla neural networks that increases its representational capacity with more layers and more hidden units per layer was the antecedent for the investigation of stacked recurrent architectures. As we previously said in section 2, stacked recurrent architectures outperform single-layer networks on multiple tasks such as phoneme recognition [8].

A stacked recurrent architecture can be defined as a deep learning model comprised of multiple recurrent layers where the output of one layer serves as the input to a subsequent layer. The intention of stacking recurrent layers is to increase the representation power of a single recurrent neural network in order to induce representations at differing levels of abstraction across layers [11]. Critically important in this stacked architecture is how a recurrent layer provides a full sequence output rather than a a single value output obtained at the last timestep. Figure 1 shows the structure of a stacked recurrent architecture.

Now we describe the kind of recurrent layers to be stacked in the definition of a deep learning model, that is, the kind of recurrent neural networks specialized for processing a sequence of elements where the order of the textual elements (e.g.

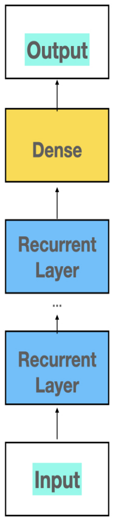

Fig. 1. Stacked Recurrent Architecture

words) is relevant to understand a document. Since recurrent neural networks can be built in many different ways, we study three basic and popular architectures: simple recurrent networks (SimpleRNN), LSTM and GRU.

Each of these architectures was created to address some linguistic concern. Simple recurrent networks (SimpleRNN), also known as Elman Networks [5], are the first neural architecture to represent the temporal nature of language as each element of a sequence is processed at a time. The key point in this model is the computation of the hidden layer: to activate the current hidden layer is necessary the value obtained in the previous hidden layer corresponding to a preceding point in time.

Equation (1) expresses the computation of the hidden layer ***h*** where ***x*** denotes the sequence (i.e. input) and ***g*** an activation function. ***W*** denotes the weight matrix corresponding to the input x_t whereas ***U*** denotes the weight matrix corresponding to the hidden layer of the previous timestep h_{t-1}. In this way, this connectionist model is concerned with the context corresponding to each element of the sequence:

$$h_t = f(U h_{t-1} + W x_t). \tag{1}$$

However, since only previous elements are taken into consideration, SimpleRNN cannot keep track of long-term dependencies.

Long Short–Term Memory (LSTM) networks were created to include the consideration of distant constituents and, in this way, to extend the local context to be analyzed. Vanishing gradient problem is another difficulty with SimpleRNNs that arises during the backward process for updating the weights. In order to overcome these problems, LSTM networks were created with three more gates to forget information that is no longer needed and to add information for posterior decisions [9].

It is precisely the addition of these gates that makes of LSTM a complex recurrent network: 4 gates and 2 weights (U and W) to learn for each gate. As an alternative to the LSTM network, the Gated Recurrent Unit (GRU) was created by [3]. By reducing the number of gates to only 2 and removing the context vector, GRU was introduced as an architecture so effective as LSTM but less complicated. Goldberg mentions, in his analysis of the multiple neural network models for NLP [6], the work of comparison between LSTM and GRU carried out by Chung et al [4]. This investigation evaluated these two recurrent architectures on two tasks and found how the performance of GRU was comparable to LSTM.

3.2 Linguistic Characteristics and Complexity

The linguistic characteristics for the analysis of how complex a text is are explained here. As a fundamental unit in the collection of texts, a sentence and its complexity are crucial in this investigation. Since the complexity of a sentence is inherent to the difficulty of its understanding, we analyze the complexity of a sentence from the perspective of the computational difficulties involved in its processing. To be more precise, we analyze the complexity of a sentence from the perspective of the computational difficulties involved in the processing of the diverse recurrent neural architectures.

A review of the literature on linguistic complexity lead us to the work of Miller and Chomsky who showed how a embedded structure, that is, a syntactic structure that contains another syntactic structure nested within itself, is particularly difficult for understanding and parsing processing [17].

In children education, an important area of assessment is the understanding of complex sentences [21]. Since a complex sentence is mainly used for expressing a more elaborate thought, it is crucial the use and identification of complex sentences. In this study, the identification of a complex sentence is based on the definition of R. Paul [19]: a complex sentence is an embedded sentence (it contains an independent clause and a dependent clause) or conjoined sentence (it contains two or more independent clauses joined together using a conjunction).

In addition to the analysis of complexity based on the number of embeddings, we also explore the use of entropy, as how it is well known, entropy is a measure of information randomness. We explore in this work the use of entropy of a text as a metric of how rich the vocabulary of the text is. Taking as reference the work of [12] in which he makes evident a correlation between the entropy of a sentence and the complexity required for its comprehension, the consideration of entropy in this work is based on the assumption that if the text is more complex, the author uses a more varied vocabulary. We represent a text as a sequence of words W = $\{w_1, w_2, \ldots, w_n\}$ to compute the entropy as follows:

$$H(w_1, w_2, \ldots, w_n) = -\sum_{i=1}^{n} p(w_i) \log p(w_i). \quad (2)$$

Finally, we also contemplate textual properties for the analysis of how long a text is. Based on the assumption that if the text is more argumentative, the author makes use of more words and sentences, the number of sentences and the number of content words in the text are analyzed to determine the impact on the performance of a learning model [13].

4 Experimental Evaluation

In order to provide empirical evidence for the relationship between different linguistic characteristics of the texts and performance of stacked recurrent architectures, our experimentation is based on text classification.

In particular, we focus our attention on the identification of constructive comments. In this way, we begin this section with the description of the corpus, that is, the collection of constructive and non-constructive comments used in the experimentation. Then, we describe the framework of the experimentation and conclude with the presentation of the obtained results for each particular linguistic characteristic and each deep learning model.

4.1 Data

The corpus used in our work, known as Constructive Comments Corpus (C3), is a collection of 12,000 online news comments with metadata information about constructiveness and toxicity [14].

The distribution of comments to classes represents an almost balanced dataset: 6,516 constructive comments (54%) and 5,484 non-constructive comments (46%). The comments to be submitted to the annotation process were obtained from the SFU Opinion and Comments Corpus (SOCC) which contains a collection of opinion articles and the comments posted by readers in response to the article [15].

What does constructive mean? To give answer to this question, a set of characteristics was defined for the concept of constructive as well as a set of characteristics for the notion of non-constructive. The sub-characteristics corresponding to each concept are shown in Table 1.

In this way, when the comment exhibits properties as evidence and dialogue there is a high probability that annotators classify the comment as constructive. And the opposite is also possible: when the comment exhibits properties as irrelevant and sarcastic there is a high probability that annotators classify the comment as non-constructive.

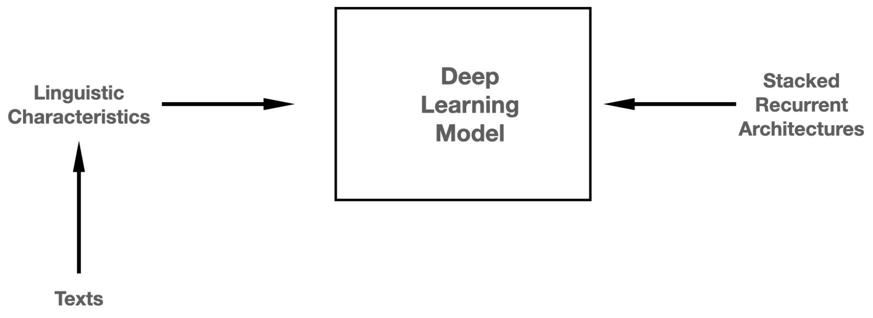

Fig. 2. Experimentation process

Table 1. Characteristics

	sub-characteristic
Constructive	constructive
	dialogue
	solution
	specific_points
	personal_story
	evidence
Non-Constructive	non_constructive
	provocative
	sarcastic
	no_respect
	unsubstantial
	non_relevant

4.2 Experimental Setup

The Figure 2 illustrates the experimentation process: from the comments collection we extract a subset according to a particular linguistic characteristic, and then, the extracted subset is provided to the deep learning model corresponding to each of the stacked recurrent architectures to be considered in this study.

Before displaying the results for each linguistic characteristic, we introduce the setup for the definition of the deep learning models. The first layer of each deep learning model is defined by learning a distributed representation corresponding to the subset of complex comments previously extracted.

In other words, we define a deep learning model with a word embedding as input where a word is represented with an output vector size of 32.

The stack is created by progressively increasing the recurrent layers from 1 to 5 so we can analyze the impact of systematically increasing the layers in the performance of the learning model.

The model is trained by using Adam as the stochastic gradient descent optimizer for 10 epochs. Additional parameters for training are: batch size = 64, learning rate = 0.01, and loss function = 'binary crossentropy'.

In this way, a deep learning model is defined for each recurrent neural architecture analyzed in this work: simple recurrent network, LSTM and GRU. As the number of available comments for each linguistic characteristic varies, we perform three-fold cross validation to use all of the pertinent comments in the subset.

4.3 Experimental Results

Embedded Clauses: in this case we first extract the subset of comments which contains embedded or conjoined sentences (from 0 to 3). The obtained results for each deep recurrent architecture are shown in Figure 3 where we can see how the performance of the SimpleRNN model improves as the number of layers increases regardless the number of clauses the text contains. On the other hand, increasing the layers to the GRU model doesn't influence the initial performance except when the text contains three clauses at least. However, the best performance is obtained with the LSTM model when the comments contains two clauses at least, that is, texts of great linguistic complexity, and the number of layers is increased.

Entropy: in this case we first extract the subset of comments for each entropy value (from 1 to 4) as a useful indicator of how rich the vocabulary of the text is. The obtained results for each deep recurrent architecture are shown in Figure 4 where we can see how the best performance is obtained with the SimpleRNN model regardless the entropy value. By increasing the number of layers, the SimpleRNN model proves to be able to cope with lexical variation. On the other hand, increasing the layers to the LSTM and GRU models doesn't cause any impact on the initial performance.

Number of sentences and words: in this case we first extract a subset of comments for each limit of sentences (from 1 to 4) and for each limit of words (from 10 to 40) as an indicator of the computational processing demanded by the use of more sentences and words. The obtained results for each deep recurrent architecture are shown in Figure 5 and 6.

About the impact of the number of sentences, in Figure 5 we see how the performance of the SimpleRNN model improves as the number of layers increases regardless the number of sentences the comment contains. On the other hand, increasing the layers to the GRU model doesn't influence the single-layer performance except when the text contains four sentences at least. However, the best performance is clearly obtained with the LSTM model when the linguistic complexity is high, that is, when comments contains three sentences at least. Said in another way, when the comment substantiates its claims by providing more details or reasons, increasing the number of recurrent layers notoriously improves the performance of the LSTM model.

Now, considering the number of words, in Figure 6 we see how, by increasing the number of layers, the best performance is obtained with the SimpleRNN model except when the comment contains 40 words at least. However, the best performance is obtained with the LSTM model when the comments contains 40 words at least, that is, by increasing the number of layers, the LSTM model is able to cope with long and more complex texts. On the other hand, increasing the layers to the GRU model doesn't influence the initial performance so a shallow model seems to be a good option.

5 Discussion

The benchmark for a discussion is the obtained results by the creators of the Constructive Comments Corpus known as C3 [14]. They implemented multiple experiments with two different learning models: a classic model based on predetermined features and deep learning models. Multiple sets of features were analyzed with a classic SVM model in order to figure out what properties cause high impact in predicting constructiveness. In our discussion, we make reference to those features related to the linguistic characteristics analyzed in this work only.

Embedded clauses: to have a better analysis of the impact of the linguistic complexity denoted by the number of embedded clauses in the texts, Figure 7 shows how each stacked recurrent architecture behaves in different levels of complexity. We see how all recurrent architectures struggle when the complexity increases. Thus, this is an empirical evidence of how the complexity of a text is related to the number of embeddings, that is, the number of nested clauses within the sentence [19].

Now, which has been the impact of a stacked architecture? We see how by increasing the number of recurrent layers the best performance is obtained with a SimpleRNN model when the level of complexity is low (stack size = 2). On the

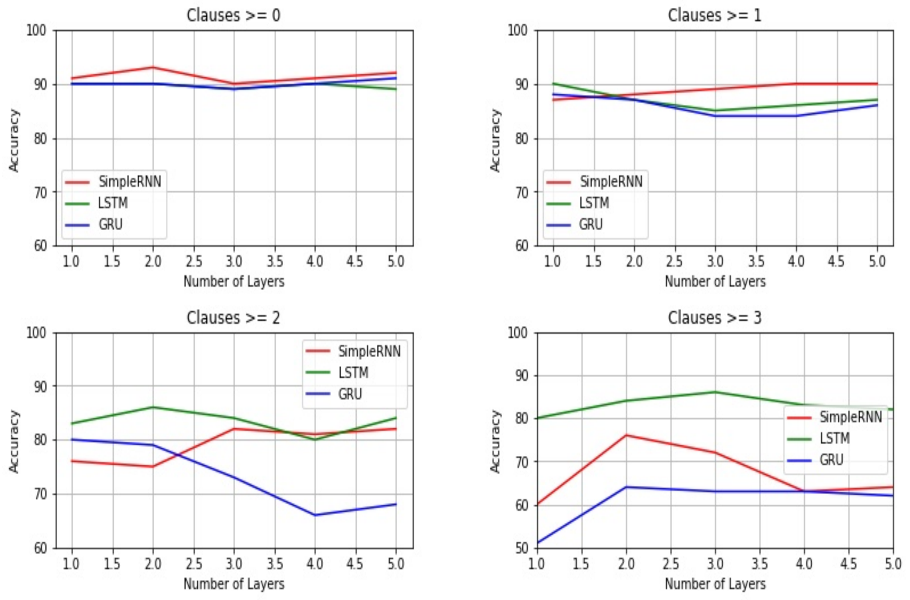

Fig. 3. Comment's clauses and stacked recurrent architectures

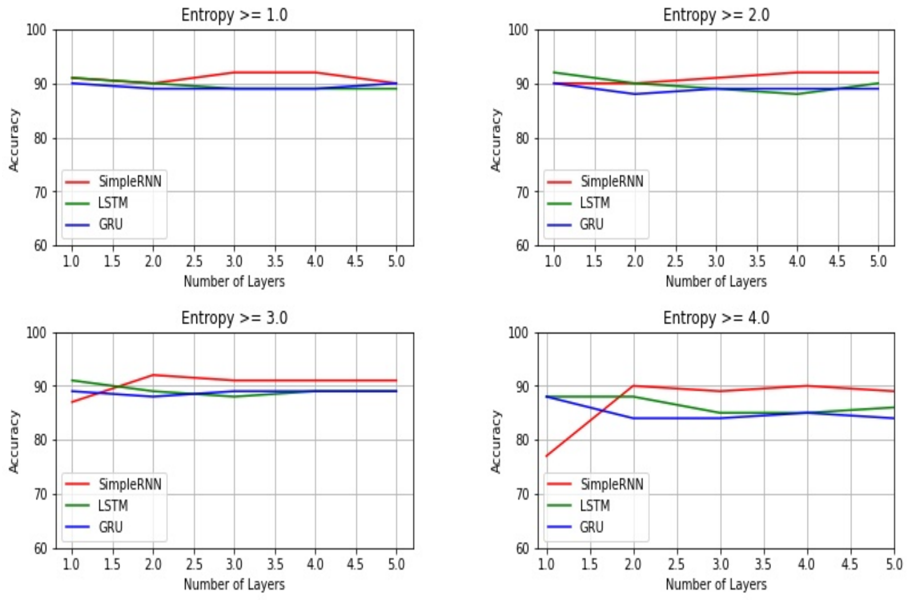

Fig. 4. Comment's entropy and stacked recurrent architectures

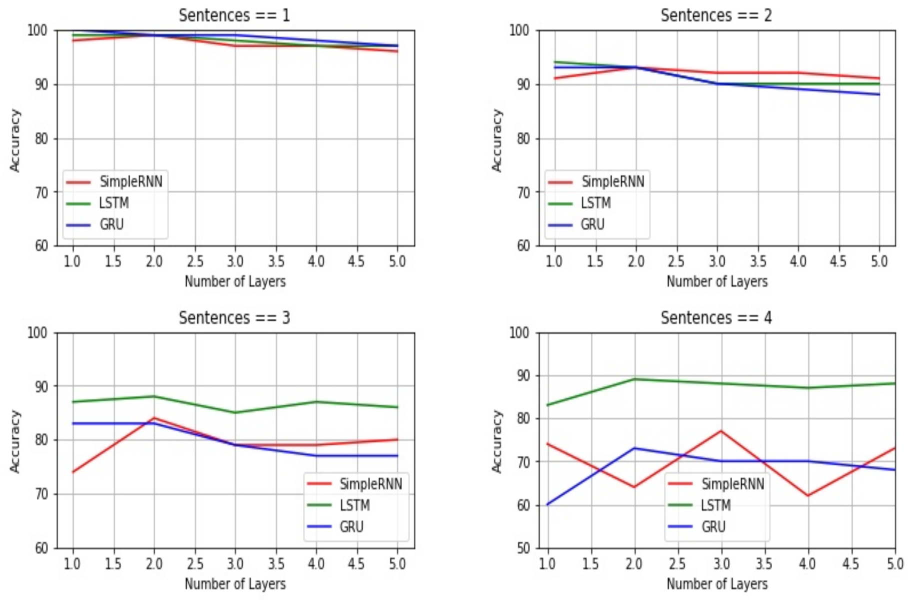

Fig. 5. Comment's sentences and stacked recurrent architectures

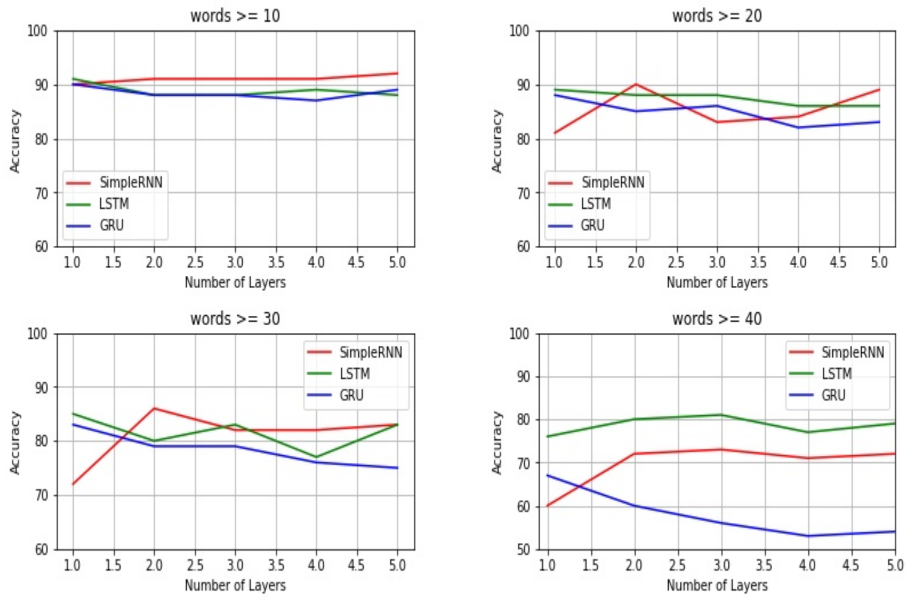

Fig. 6. Comment's words and stacked recurrent architectures

other hand, when the level of complexity is high, the best performance is obtained with a LSTM by increasing the number of layers (stack size = 3). Thus, stacked recurrent architectures prove to be useful to cope with high and low values of this particular linguistic complexity.

We now compare the performance of our models with the results obtained by Kolhatkar et al. [14]. They implemented a classic SVM model with multiple sets of features where *argumentation* is the set of features comparable to the complex sentences analyzed in our case. The set of *argumentation* features is:

— presence of discourse connectives (therefore, due to),

— reasoning verbs (cause, lead),

— abstract nouns (problem, issue, decision, reason),

— stance adverbials (undoubtedly, paradoxically).

These features were selected based on the assumption that an argumentative text is one that exhibits reasons and explanations. The result obtained by the SVM model was a 0.76 F1 score whereas ours results for different levels of complexity are shown in Figure 7. As we can see in Table 2 (the values are obtained from Figure 7), when the comments contains one or two complex sentences at least, the results obtained by the recurrent architectures are higher than the SVM model.

Table 2. Two embedded clauses and stacked recurrent architectures

model	stack size	acc
SimpleRNN	3	0.82
LSTM	2	0.86
GRU	1	0.80

However, when the comments contains three or more complex sentences, the result obtained by SimpleRNN is identical to the SVM model whereas the result obtained by GRU is lower.

Table 3. Three or more embedded clauses and stacked recurrent architectures

model	stack size	acc
SimpleRNN	2	0.76
LSTM	3	0.86
GRU	2	0.64

Table 3 shows a stacked LSTM model clearly able to cope with the classification of argumentative comments, that is, the highest level of linguistic complexity.

Entropy: Figure 8 shows the impact of the linguistic complexity denoted by the entropy of the texts where different levels of entropy for each stacked recurrent architecture is displayed. In this case we also see how the performance of all recurrent architectures drops when the complexity increases. For this reason the consideration of textual entropy as linguistic complexity makes evident the correlation between the entropy of a sentence and the complexity required for its comprehension suggested by Keller [12].

About the impact of a stacked architecture we see in Figure 2 how by increasing the best performance is obtained with a SimpleRNN model for any level of complexity. For example, when the level of complexity is low (entropy = 2), a stack of four recurrent layers obtains good performance; and when the level of complexity is high (entropy = 4), a stack of four recurrent layers outperform the LSTM and GRU architectures. On the other hand, a deep model based on LSTM and GRU architectures had no positive effect so a shallow model was the best option in these cases.

In order to compare the performance of our models with the results obtained by Kolhatkar et al. [14], *Text quality* is the set of features comparable to the use of entropy as an indicator of a more varied vocabulary. The set of *Text quality* features is:

— Readability score,

— Personal experience score.

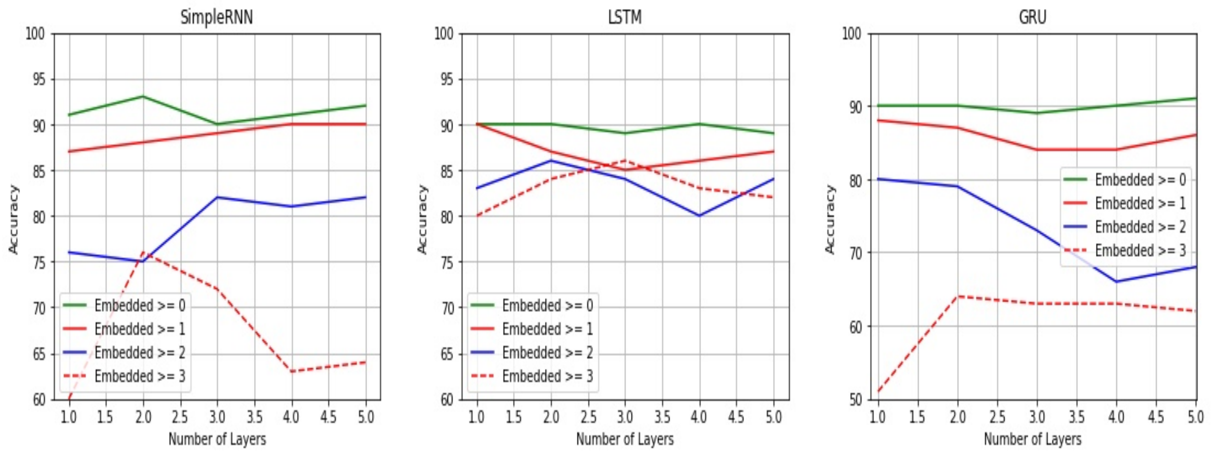

Fig. 7. Stacked recurrent architectures and Embedded clauses

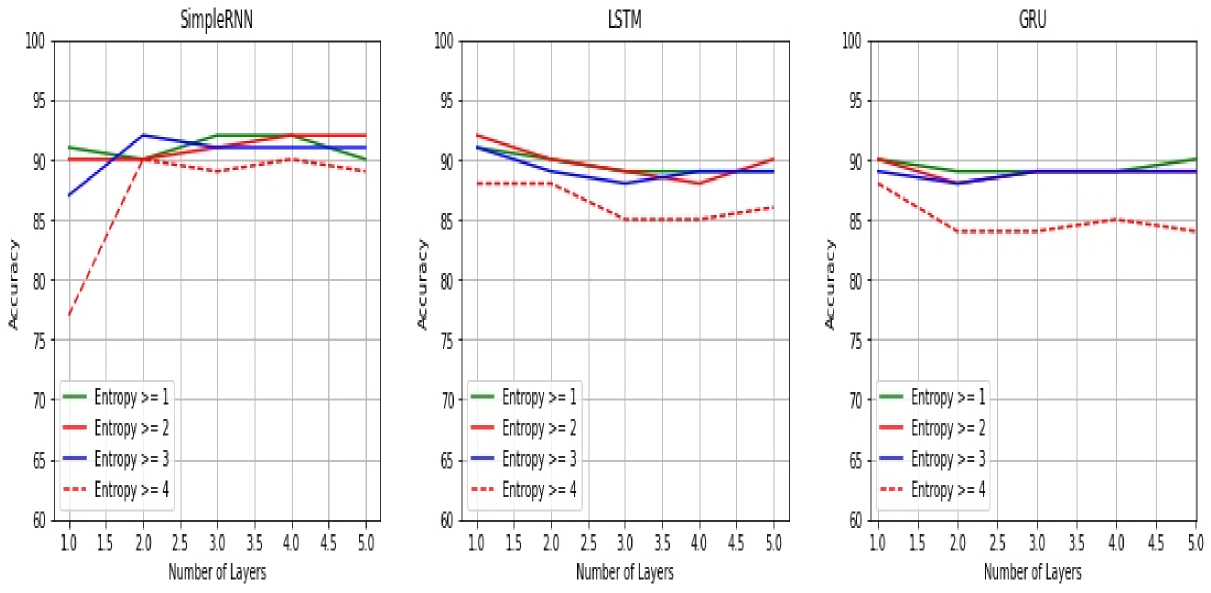

Fig. 8. Stacked recurrent architectures and Entropy values

These features were selected as an attempt to quantify how hard a text is to read. The result obtained by the SVM model was a 0.90 F1 score whereas ours results for different entropy values are shown in Figure 8. As we can see in Table 4 (the values are obtained from Figure 8), when the comments contains a low entropy value ($entropy \geq 2$), the results obtained by the deep recurrent architectures are higher than the SVM model.

However, when the comments contains a high entropy value ($entropy \geq 4$), the result obtained by SimpleRNN is identical to the SVM model whereas the results obtained by LSTM and GRU are lower. Table 5 shows a stacked SimpleRNN model achieving similar performance to the SVM model when the readability of the comments demands a rich vocabulary, that is, when the comments exhibit the highest level of linguistic complexity.

As we previously said, Table 4 and Table 5 also show how a deep model based on LSTM and GRU architectures had no positive effect so a shallow model was the best option in both cases.

Number of sentences and words: in this case, from the results obtained by the analysis of number of sentences and words, we can see in Figure 9 and Figure 10 how precision decreases as the number of sentences and words in the comments increases. This decline in the performance of the recurrent architectures is a clear-cut evidence that the complexity of constructive comments is related to the number of sentences and words.

Figure 9 and Figure 10 also illustrates the impact of a stacked SimpleRNN architecture. In fact, we see how by increasing the number of recurrent layers the best performance is obtained with a SimpleRNN model when the level of complexity is low (stack size = 2 for sentences and words). Now, when the level of complexity is high, the best performance is obtained with a stacked LSTM architecture: stack size = 2 for sentences and stack size = 3 for words. On the other hand, a deep model based on GRU architecture had minimum impact on the classification task.

In order to compare the performance of our models with the results obtained by Kolhatkar et al. [14], *Length* is the set of features comparable to the use of number of sentences and words as an indicator of a constructive comment. The set of *Length* features is:

— Number of tokens in the comment,

— Number of sentences,

— Average word length,

— Average number of words per sentence

These features were selected to verify the premise that the length of the text is a good predictor of constructiveness. The result obtained by the SVM model was a 0.93 F1 score whereas ours results for various number of sentences and words are shown in Figure 9 and Figure 10. As we can see in Table 6 (the values are obtained from Figures 9 and 10), when the comment is short (number of sentences = 2), the results obtained by a deep SimpleRNN architectures and GRU are identical to the SVM model. The performance of a single layer LSTM network is a bit higher on this level of complexity.

Table 4. Low entropy ($entropy \geq 2$) and stacked recurrent architectures

model	stack size	acc
SimpleRNN	4	0.92
LSTM	1	0.92
GRU	1	0.90

Table 5. High entropy ($entropy \geq 4$) and stacked recurrent architectures

model	stack size	acc
SimpleRNN	4	0.90
LSTM	1	0.88
GRU	1	0.88

Table 6. Low number of Sentences and Words

	$Sentences = 2$		$Words \geq 20$	
model	stack size	acc	stack size	acc
SimpleRNN	2	0.93	2	0.90
LSTM	1	0.94	1	0.89
GRU	1	0.93	1	0.88

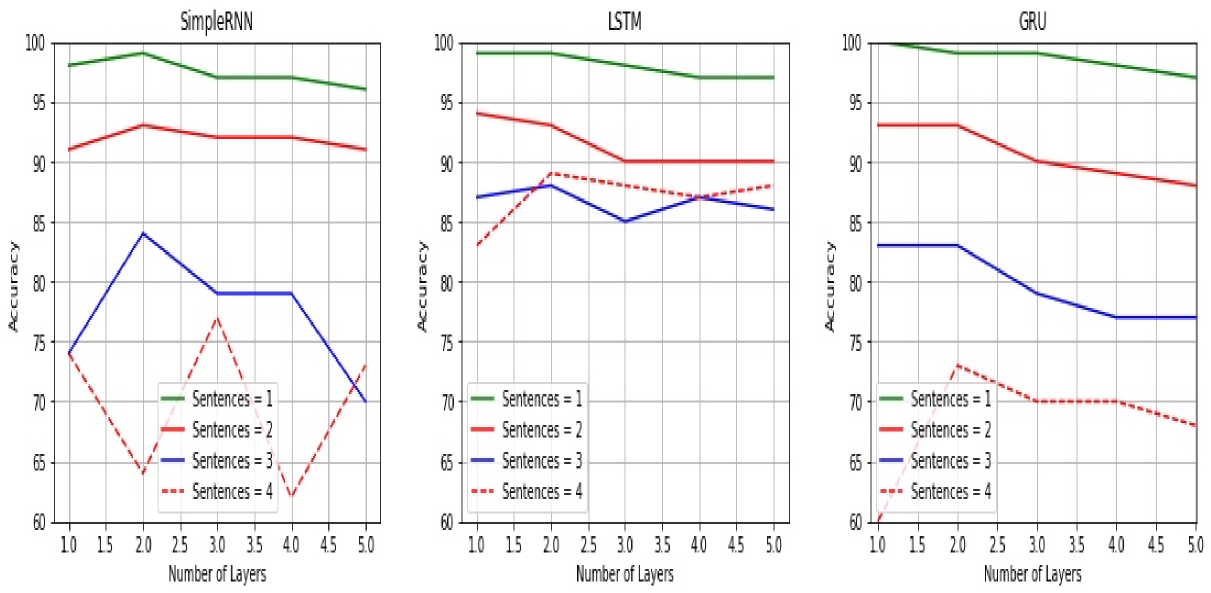

Fig. 9. Stacked recurrent architectures and Sentences values

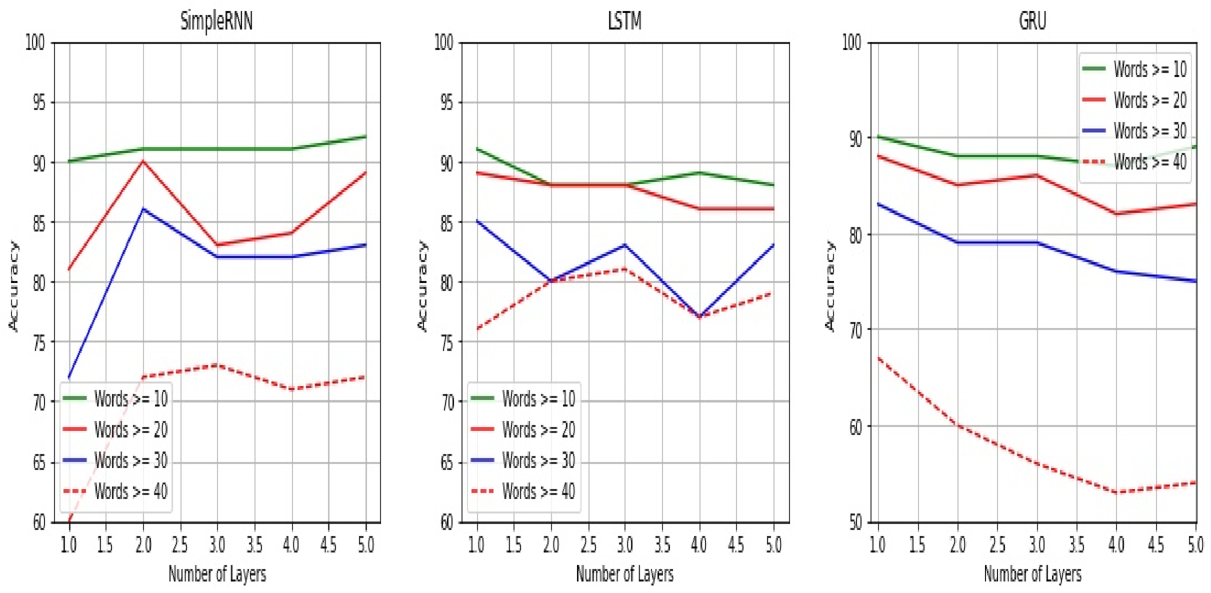

Fig. 10. Stacked recurrent architectures and Words values

Now, when the linguistic complexity of the text is high, that is, when long comments contain at least 4 sentences or at least 40 words, a stacked model implemented with a LSTM architecture obtains the best performance. Table 7 shows how a stacked LSTM model outperforms the SimpleRNN and GRU architectures.

Table 7. High number of Sentences and Words

	$Sentences = 4$		$Words \geq 40$	
model	stack size	acc	stack size	acc
SimpleRNN	3	0.77	3	0.73
LSTM	2	0.89	3	0.81
GRU	2	0.73	1	0.67

Computational cost: although increasing the representational power of the network, stacking recurrent layers entails a tradeoff to be considered. In fact, stacking recurrent layers generates a tradeoff between increasing network capacity and demanding higher computational resources. Since determining the runtime and memory requirement of the recurrent architectures is highly platform-dependent, we do not describe in this work the computational cost in absolute terms. We describe rather the computational cost as a degree of runtime.

In this way, we show evidence of the computational cost of a stack recurrent architecture from two perspectives:

— Processing time required for multiple number of recurrent layers,

— Processing time required for each recurrent architecture.

Figures 11 and 12 display the percent of processing time required for each number of recurrent layers considered in this investigation: from 1 to 5 layers. Figure 11 shows the computational resources demanded by a LSTM model processing texts that contain one embedded clause at least whereas Figure 12 shows the same LSTM model processing texts that contain two sentences. A linear correlation between depth and time is observed in both figures: the computational processing increases as the depth of the model increases.

Now, Figures 13 and 14 display the percent of processing time required for each recurrent architecture considered in this investigation: SimpleRNN, LSTM and GRU.

Figures13 and 14, corresponding to the processing of texts that contain one embedded clause at least and texts that contain two sentences respectively, show how a stacked learning model based on the GRU architecture proves to be the most demanding model. On the other hand, a deep model based on the primitive SimpleRNN architecture does not demand substantial computational resources.

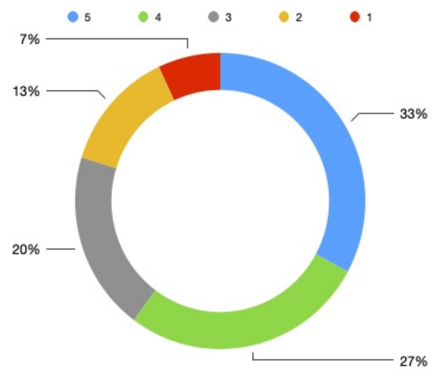

Fig. 11. LSTM model processing texts that contain one embedded clause

6 Conclusion and Future Work

We have analysed in this paper the implications of linguistic complexity characteristics in the performance of a deep learning model defined in terms of a stacked recurrent architecture. To be more specific, we explore linguistic characteristics for the analysis of how complex a text is and, in this way, to investigate the relationship between the linguistic complexity of the texts and the depth of the learning model.

By using qualitative (embedded clauses and entropy) and quantitative (number of sentences and words) methods, our experimentation based

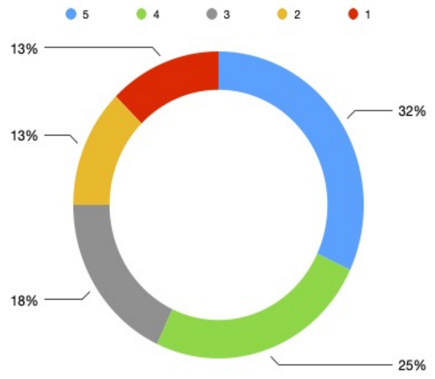

Fig. 12. LSTM model processing texts of two sentences

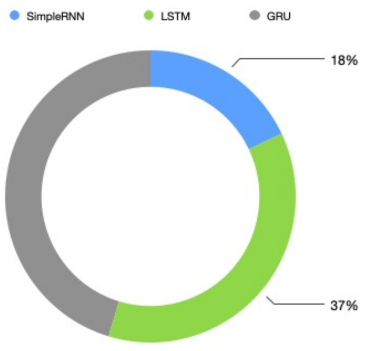

Fig. 13. LSTM model processing texts that contain one embedded clause

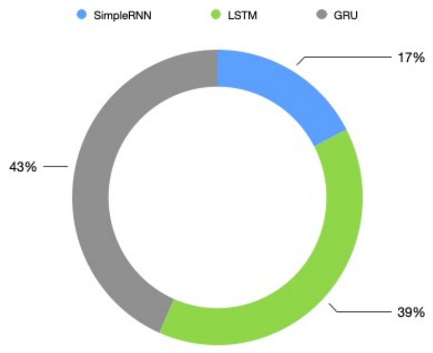

Fig. 14. LSTM model processing texts of two sentences

on the classification of constructive comments provides empirical evidence of how the linguistic complexity characteristics of the comments impact the stacked recurrent architecture for the identification of constructive comments.

For example, when the number of complex sentences in the comments increases, a single-layer recurrent architecture struggle on the identification of constructiveness. Something similar occurs when the entropy in the comments increase. We show how by increasing the number of recurrent layers, that is, by implementing a stacked recurrent architecture, a better performance is achieved.

In fact, by applying stacked recurrent architectures based on deep learning algorithms such as SimpleRNN, LSTM and GRU, it was posible to observe both the depth of the model improving the results and the number of layers for a model to be lost or stuck (i.e. overfitting) during the optimization process. Finally, we also show how increasing the representational power of the network by stacking recurrent layers entails a tradeoff between increasing network capacity and demanding higher computational resources.

About future work, we are interested in exploring different datasets. The experimentation conducted exhibited how the number of comments available for identification of constructiveness decreases when the linguistic complexity increases. In order to extend our conclusions, it is necessary the use of a different corpus that allows to put aside this limitation.

We are also interested in exploring the intuition that adding a custom attention layer to recurrent neural networks can improve their performance when the linguistic complexity of the texts increases. Since adding attention component to the network has shown significant improvement in tasks such as text summarization and machine translation [7], we want to investigate how an attention component contribuyes to a better representation of complex texts (which contain embedded clauses and long sentences) for classification tasks.

References

1. **Bengio, Y. (2009).** Learning deep architectures for ai. Foundations and Trends in Machine Learning, Vol. 2, No. 1, pp. 1–127.

2. **Bengio, Y., LeCun, Y., Hinton, G. (2021).** Deep learning for ai. Communications of the ACM, Vol. 64, No. 7, pp. 58–65.

3. **Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y. (2014).** Learning phrase representations using RNN encoder–decoder for statistical machine translation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, pp. 1724–1734.

4. **Chung, J., Gulcehre, C., Cho, K., Bengio, Y. (2014).** Empirical evaluation of gated recurrent neural networks on sequence modeling. NIPS 2014 Deep Learning and Representation Learning Workshop, pp. 1–9.

5. **Elman, J. L. (1990).** Finding structure in time. Cognitive Science, Vol. 14, No. 2, pp. 179–211.

6. **Goldberg, Y. (2016).** A primer on neural network models for natural language processing. Journal of Artificial Intelligence Research, Vol. 57, pp. 345–420.

7. **Goodfellow, I., Bengio, Y., Courville, A. (2016).** Deep Learning. MIT Press.

8. **Graves, A., rahman Mohamed, A., Hinton, G. E. (2013).** Speech recognition with deep recurrent neural networks. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 6645–6649.

9. **Hochreiter, S., Schmidhuber, J. (1997).** Long short-term memory. Neural Computation, Vol. 9, No. 8, pp. 1735–1780.

10. **Jain, P., Agrawal, P., Mishra, A., Sukhwani, M., Laha, A., Sankaranarayanan, K. (2017).** Story generation from sequence of independent short descriptions. arXiv, Vol. 1707.05501, pp. 1–7.

11. **Jurafsky, D., Martin, J. H. (2022).** Speech and language processing. 3rd ed. To be published.

12. **Keller, F. (2004).** The entropy rate principle as a predictor of processing effort: An evaluation against eye-tracking data. Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Barcelona, Spain, pp. 317–324.

13. **Kolhatkar, V., Taboada, M. (2017).** Using new york times picks to identify constructive comments. Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism,, Copenhagen, Denmark, pp. 100–105.

14. **Kolhatkar, V., Thain, N., Sorensen, J., Dixon, L., Taboada, M. (2020).** Classifying constructive comments. arXiv, Vol. 2004.05476, pp. 1–24.

15. **Kolhatkar, V., Wu, H., Cavasso, L., Francis, E., Shukla, K., Taboada, M. (2019).** The sfu opinion and comments corpus: A corpus for the analysis of online news comments. Corpus Pragmatics, Vol. 4, pp. 155–190.

16. **Krizhevsky, A., Sutskever, I., Hinton, G. (2012).** Imagenet classification with deep convolutional neural networks. Proceedings of NIPS'2012, Curran Associates, Inc., pp. 1–9.

17. **Miller, G. A., Chomsky, N. (1963).** Finitary models of language users, volume II, chapter Handbook of Mathematical Psychology. John Wiley & Sons, pp. 419–491.

18. **Pascanu, R., Gulcehre, C., Cho, K., Bengio, Y. (2014).** How to construct deep recurrent neural networks.

19. **Paul, R. (1981).** Analyzing complex sentence development, chapter 2. University Park Press, pp. 36–71.

20. **Santucci, V., Santarelli, F., Forti, L., Spina, S. (2020).** Automatic classification of text complexity. Applied Sciences, Vol. 10, No. 20, pp. 1–19.

21. **Steffani, S., Dachty, L. (2007).** Identifying embedded and conjoined complex sentences: Making it simple. Contemporary Issues in Communication Science and Disorders., Vol. 34, No. 2, pp. 44–54.

22. **Utgoff, P. E., Stracuzzi, D. J. (2002).** Many–layered learning. Neural Computation, Vol. 14, pp. 2497–2539.

23. **Wang, X., Liu, Y., Sun, C., Wang, B., Wang, X. (2015).** Predicting polarities of tweets by composing word embeddings with long short-term memory. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the

7th International Joint Conference on Natural Language Processing, Association for Computational Linguistics, Beijing, China, pp. 1343–1353.

24. **Yasseri, T., Kornai, A., Kertész, J. (2012).** A practical approach to language complexity: A wikipedia case study. PLOS ONE, Vol. 7, No. 1, pp. 1–8.

Article received on 02/10/2021; accepted on 19/01/2022. Corresponding author is Diego Uribe.