Extraction, Attribution, and Classification of Quotations in Newspaper Articles

João Godinho

Abstract—Reported speech is a crucial part of news articles, which frequently rely on quotations to report on the perspectives and opinions of direct participants in the narrated events. The ability to accurately extract and organize these quotations is highly relevant for text mining applications aiming to reduce human intervention in media monitoring, help journalists in fact-checking, or aid media scholars and general users to browse the news. Several previous studies have addressed the extraction of reported speech from news articles, although often addressing only direct quotations and/or using relatively simple methods relying heavily on hand-crafted features. This article extends these previous studies in several directions, evaluating the application of modern deep learning methods for quotation extraction (i.e., for delimiting occurrences of direct, indirect, and mixed quotations), attribution (i.e., for assigning quotations to the corresponding authors, as mentioned in the surrounding text), and classification (i.e., for assigning quotations to numerical scores encoding emotional valence and intensity). Particularly innovative aspects include the use of Nested-LSTMs, as opposed to more common RNNs, or the association of quotations to emotional valence and intensity. Experimental results show that relatively simple neural architectures, based on RNNs, can achieve very good results in all three aforementioned tasks, outperforming previously reported results.

Index Terms—Text Mining, Media Monitoring, Deep Learning for NLP, Mining Reported Speech, Emotion Detection

1 Introduction

Reported speech is a crucial part of news articles, which commonly make use of quotations to support the claims and perspectives identified by the journalists. Through the inclusion of quotations from experts and/or observers (i.e., witnesses or individuals involved in the reported events), news articles are made more concrete and more personal. Given the ubiquity in the use of reported speech, the ability to accurately extract and organize quotations in newspaper articles is highly relevant for text mining applications aiming to reduce human intervention in media monitoring, help journalists in fact-checking, or aid media scholars and general users to browse and keep track of relevant news (e.g., in the context of applications supporting the analysis of what politicians say about particular issues, and how this changes over time).

Over the last few years, many authors have already proposed Natural Language Processing (NLP) methods for extracting quotations and attributing them to the corresponding individuals [1, 2, 3, 4]. However, most previous studies considered only direct quotations, and they have relied on relatively simple approaches (e.g., most previous efforts relied on rule-based methods [5, 6], for instance based on searching for text between quotation marks, and attributing the quotation to the nearest person name reference, if a reporting verb such as *say* occurs in between). Complex instances of reported speech, such as indirect quotations (i.e., not involving the orthographic cue of quotation marks, such as in the example *Despite this loss, First Chicago said it does not need to sell stock to raise capital*) or mixed quotations (i.e., combining characteristics of both direct and indirect quotations, such as in the example *White House officials said Bush is "fully satisfied" with CIA Director Webster and the agency's performance during the coup in Panama*), have mostly been ignored in previous text mining efforts. Moreover, apart from few exceptions [1, 7, 8], most previous studies have ignored tasks related to the classification of the extracted quotations. Given that quotations are used in news articles as evidence of a person's opinions with respect to particular subjects, it may be interesting to classify quotations according to the opinions that they express.

This article extends previous studies in several directions, evaluating the application of modern deep learning methods for quotation extraction (i.e., for delimiting the occurrences of direct, indirect, and mixed quotations), attribution (i.e., for assigning quotations to the corresponding authors, as referenced in the surrounding text), and classification (i.e., for assigning quotations to numerical scores encoding emotional valence and intensity).

Quotation extraction is specifically modeled as a sequence labeling problem (i.e., each word in a sentence is labeled according to whether it belongs to a quotation), and I propose to address the task through Recurrent Neural Network (RNN) architectures similar to those used in other NLP problems, for instance by combining bidirectional Long Short-Term Memory (LSTM) networks with Conditional Random Fields (CRFs) [9]. Quotation attribution and classification are, in turn, modeled as classification problems that take sequences of words as input. In the case of quotation attribution, I start by gathering candidate authors (i.e., the three person references, in the surrounding text, that appear closer to the quotation) and take the sequences of words encompassing the reference and the quotation, with all words in between, as input to a RNN that decides if the person reference corresponds to the author that should be attributed. The highest scoring reference is finally assigned. As for quotation classification, my approach takes inspiration on recent studies using deep learning to assign short texts to numeric ratings for emotional valence and arousal [10, 11, 12, 13], as defined in Russel's circumplex model [14]. I specifically leverage an RNN, similar to that used for quotation attribution, to assign quotations to two numeri-

• *João Godinho*
E-mail: joaodfgodinho@tecnico.ulisboa.pt

Manuscript received April 15, 2015.

cal scores, respectively encoding the dimensions of emotional valence and intensity.

The results from an extensive set of experiments, leveraging collections used in previous studies focused on mining quotations [15] or quantifying emotions [16, 17, 18] from textual contents, show that relatively simple neural architectures can achieve very good results in all three aforementioned tasks, outperforming previously reported results. Particularly innovative aspects, in the work that is reported in this article, include the use of Nested-LSTMs [19] as opposed to more common RNN architectures, or the association of quotations to emotional valence and intensity.

The rest of this article is organized as follows: Section 2 introduces fundamental concepts related to deep learning for natural language processing and presents previous work related to (a) mining reported speech and (b) emotion classification from textual information. Section 3 describes the proposed methods for addressing the quotation extraction, attribution, and emotion classification tasks, detailing also their implementation. Section 4 presents the evaluation methodology, including the resources supporting the experiments and the metrics used for assessing the quality of the results, and discusses the obtained results in comparison with previous work in the area. Finally, Section 5 summarizes the main conclusions from this research, and presents ideas for future work.

2 Concepts and Related Work

This section presents the fundamental concepts and related works which have already focused on the multiple proposed tasks (i.e., extraction and attribution of quotations, and classification of emotion in textual information).

2.1 Fundamental Concepts

This section provides all the fundamental concepts that the reader will need to have in order to fully understand the rest of the document. First, there is an introduction to the different methods used to represent textual information in NLP tasks. And later there is an introduction to neural networks architectures.

2.1.1 Representing Textual Documents

Statistical methods for Natural Language Processing (NLP) typically involve representing textual information in the form of vectors. One way of representing a document is based on representing each word as a V dimensional vector, where V is the size of the vocabulary. Each position of the vector is set to 0, except for the index representing the word, which is equal to 1. This way of representing words is called a one-hot representation.

Due to the limitations of this representation (e.g., V can be very large and there is no information sharing between similar words), there has been a movement from this representation to a dense representation. In a dense representation, each word is represented as a smaller dimensional vector, often called embeddings. After the training, words with similar meanings will present correlated vectors. With these new vectors, words with less appearance can share statistical information with others which appear more often.

Different methods for producing word embeddings have been proposed, including word2vec [20], GloVe [21], or more recently FastText [22]. For instance, in the word2vec approach, there are two ways of training these embeddings: using a Continuous Bag of Words (CBOW) or using the Skip-Gram approach.

The CBOW approach aims to predict a word given its context. Given a sequence of words w_1, w_2, ... , w_{T-1}, w_T, the goal of this model is to find the target word, w_t, using the c words in front and before the central word, maximizing the following function:

$$\frac{1}{T}\sum_{t=1}^{T}\log p(w_t|w_{t-c},...,w_{t-1},w_{t+1},...,w_{t+c}) \quad (1)$$

In the previous equation, T refers to the total number of words in the sequence and c refers to the used context size.

On the other hand, the Skip-Gram approach aims to predict the context given a central word. Given the same sequence of words used in the previous model, this model tries to maximize the average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t) \quad (2)$$

In Equation 2, c is again the length of the used window. The basic skip-gram calculates the probability $p(w_{t+j}|w_t)$ using the softmax function:

$$p(w_O|w_I)=\frac{\exp({\mathbf{v}'_{w_O}}^T\mathbf{v}_{w_I})}{\sum_{w=1}^{V}\exp({\mathbf{v}'_w}^T\mathbf{v}_{w_I})} \quad (3)$$

In the previous equation, V represents the vocabulary size, $\mathbf{v}_w$ refers to the vector representation of an input word and $\mathbf{v}'_w$ represents the representation of an output word (i.e., within the context length).

2.1.2 Neural Network Models for Natural Language Processing

Neural network architectures have been largely applied to several tasks. Tasks like sentiment analysis, text translation and computer vision have benefited from the application of these architectures in relation to classical approaches [23, 24].

One of the types of neural network architectures which have been used in tasks involving natural language processing is named recurrent neural network. These architectures allow the use of arbitrarily size inputs and they have the characteristic of preserving the input's order.

This architecture takes an ordered list of vectors, $\mathbf{x_1}$, ..., $\mathbf{x_n}$ together with an initial hidden state vector $\mathbf{s_1}$, and returns an ordered list of vectors $\mathbf{y_1}$, ..., $\mathbf{y_n}$ and a list of hidden states, $\mathbf{s_1}$, ..., $\mathbf{s_n}$. These hidden states work as the memory of the network, and each is calculated based on the previous hidden state, $\mathbf{s_{t-1}}$ and the current input, $\mathbf{x_t}$, being t the index of the current input.

The simplest recurrent architecture is the Elman RNN and can be characterized as:

$$\begin{aligned}\mathbf{s_i} &= g(\mathbf{x_i}\mathbf{W^x}+\mathbf{s_{i-1}}\mathbf{W^s}+\mathbf{b})\\ \mathbf{y_i} &= \mathbf{s_i}\end{aligned} \quad (4)$$

While this model can in theory work with all the past information, in practice it suffers from vanishing gradients, which hinder the training process. To tackle this problem, two other RNN architectures have been proposed: Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU).

These architectures solve this problem by applying gates, controlling the amount of information that should pass to the next state. The LSTM [25] architecture is formalized as follows:

$$\begin{aligned}
\mathbf{s_j} &= [\mathbf{c_j}; \mathbf{h_j}] \\
\mathbf{c_j} &= \mathbf{c_{j-1}} \odot \mathbf{f} + \mathbf{g} \odot \mathbf{i} \\
\mathbf{h_j} &= \tanh(\mathbf{c_j}) \odot \mathbf{o} \\
\mathbf{i} &= \sigma(\mathbf{x_j}\mathbf{W^{xi}} + \mathbf{h_{j-1}}\mathbf{W^{hi}}) \\
\mathbf{f} &= \sigma(\mathbf{x_j}\mathbf{W^{xf}} + \mathbf{h_{j-1}}\mathbf{W^{hf}}) \\
\mathbf{o} &= \sigma(\mathbf{x_j}\mathbf{W^{xo}} + \mathbf{h_{j-1}}\mathbf{W^{ho}}) \\
\mathbf{g} &= \tanh(\mathbf{x_j}\mathbf{W^{xg}} + \mathbf{h_{j-1}}\mathbf{W^{hg}}) \\
\mathbf{y_j} &= \mathbf{h_j}
\end{aligned} \tag{5}$$

And the GRU [26] architecture which does not present a separated memory and has fewer gates is represented as follows:

$$\begin{aligned}
\mathbf{s_j} &= (\mathbf{1} - \mathbf{z}) \odot \mathbf{s_{j-1}} + \mathbf{z} \odot \tilde{\mathbf{s_j}} \\
\mathbf{z} &= \sigma(\mathbf{x_j}\mathbf{W^{xz}} + \mathbf{s_{j-1}}\mathbf{W^{sz}}) \\
\mathbf{r} &= \sigma(\mathbf{x_j}\mathbf{W^{xr}} + \mathbf{s_{j-1}}\mathbf{W^{sr}}) \\
\tilde{\mathbf{s_j}} &= \tanh(\mathbf{x_j}\mathbf{W^{xs}} + (\mathbf{s_{j-1}} \odot \mathbf{r})\mathbf{W^{sg}}) \\
\mathbf{y_j} &= \mathbf{s_j}
\end{aligned} \tag{6}$$

To represent a given sentence, there is the choice of using the last hidden state of the RNN or use all the time-steps outputs with operations like global Max-pooling or self-attention mechanisms.

Recurrent neural networks have a limitation by only considering past information from the beginning of the input to the current token. Sometimes it can be useful to use future information which is available in front of the current token. To overcome this issue a new architecture is often used, named bidirectional RNN (BiRNN), which is composed of two independent RNN, being one responsible for the forward pass and the other for the backward pass. This way, the output of the BiRNN is the concatenation of the forward and backward passes.

2.2 Related Work

This section overviews previous studies that are relevant to the context of this work. It is divided as follows: the first section introduces meaningful work related to mining reported speech while the second section presents work that focused on the extraction of emotions from textual information.

2.2.1 Quotation Extraction and Attribution

Most previous studies focused on the extraction of only direct type quotations, leveraging rule-based/regular expression approaches: if a reporting verb is present, search for quotation marks and assign this quotation span to the closest entity [5, 6].

After these, studies started to focus on extracting all possible types of quotations (i.e., direct, indirect, and mixed quotations) [27]. In this study they started by using k-nearest neighbor algorithm to build a verb-cue classifier (i.e., classifying verbs into attributional and non-attributional) which could be used to enhance the performance of both extraction and attribution models. To train this model, they used multiple features (e.g., lexical, syntactic, word relations, and sentence features). For the task of quotation extraction, they applied two approaches, e.g. token-based and constituent-based using the same set of hand-crafted features (e.g., lexical, sentence, dependency, and external knowledge features). The approach that achieved the best results was the token-based trained with a CRF, where the system tries to label each token as inside, outside, or at the beginning of a quotation.

In a following publication, one of the authors published his thesis where he explained more in-depth the attributing quotation task [1]. For this, he built two models: a binary class model where each candidate is labeled as either speaker or ¬speaker, and a positional class model that labels each candidate as pre_{dist} or $post_{dist}$ if it is before or after the quotation respectively, being *dist* the absolute distance from the quotation to the candidate. The candidates have been automatically generated by using the BBN technologies Coreference and Entity Type Corpus, which has a set of gold-standard named entities, common nouns, and pronouns. Since the used dataset (i.e., PARC) annotates the full source span (i.e., text indicating to whom the quotation is being attributed), he considers the speaker to be the first candidate that is a subspan of each source (setting this as the speaker). When there is no match he simply inserts the complete source as a candidate. He considers a token to be a candidate if it is a proper noun, common noun, or pronoun and it is within the 6 candidates before and after the quotation. For learning, the models used a maximum entropy classifier for the binary classification and a CRF for the positional model. The used features include distance, paragraph, sentences, context, quotation, candidate, category, dependency, pattern, source, and conversation features. To decode the sequence of possible attributions to the quotations he implemented several methods, including a greedy (which simply chooses the most likely candidate using past information), a Viterbi and for the positional class model he also implemented a CRF decoding method.

Later, Pareti also presented her work in the area of quotations [28] (in part common to the previous work [1]). In this work, she started by upgrading the previously used dataset to a new version (i.e., PARC3), which corrected missing annotations and added nested quotations. She proceeded to update the pre-created models (with minor changes) to this new dataset, focusing on the same tasks (i.e., cue and content detection and entity attribution). She extended previous works by (i) extracting of the complete source span and (ii) linking the content and source span to the correct cue span.

More recently, a work tried to show that the Markov assumptions used in previous studies that use CRFs [1, 27, 28] do not take into account joint decisions over the whole sentence, proposing an algorithm based on a more flexible version of the Markov model [4]. To do this, they first formalized the problem as a boundary detection task, where the goal is to find the beginning and end token instead of using a CRF which may present problems when keeping track of the complete sequence over an extended span. As well as previous works, they initially detected cues, and modeled the three tasks (i.e., cue, begin and end detection) with the following score function:

$$\text{score}_x(t) = \mathbf{w}_x \mathbf{f}_x(t) \tag{7}$$

Where in the last equation, t represents the token, class $x \in \{c, b, e\}$, $\mathbf{f}(t)$ is the feature extractor and $\mathbf{w}$ is the weight vector which is estimated using perceptron algorithm.

To prove their ideas, they initially created a greedy model that for each cue, adds all spans within a maximum distance (from the cue) which length is less than a certain limit. If the candidate is overlapping any other span, the algorithm removes it. The features used in this work were mostly derived from previous study [28].

Their complete system (i.e., SemiMarkov) uses a relaxed Markov architecture, able of handling global features regarding the complete span. It begins by using possible quotations (constituted by beginning and end tokens) that are given by two probability distributions: P_b and P_e. Using Equation 7 to calculate the score, these distributions are defined as:

$$P_x(t) \propto \exp\left(\frac{\text{score}_x(t)}{T_x}\right) \tag{8}$$

Where in the previous equation, x can be either the begin or end token, and T_x represents a temperature hyperparameter used to encourage the selection of lower score tokens. Having now a candidate, (t_b, t_e), they can use global features (e.g., is there a token that is classified as a cue in a 5 token window, and distance to the previous and next cue). The next step of the algorithm is to verify if the candidate's score is greater than the sum of scores of all spans overlapping it. If yes, it accepts the candidate and removes overlapping spans. Else, the candidate is simply rejected. It is also trained using the perceptron algorithm. They tested their system (i.e., SemiMarkov) in the PARC3 dataset where it outperformed the state-of-the-art on every type of quotations.

2.2.2 Classifying Text with basis on Emotion Analysis

In a study addressing the analysis of emotions expressed in Facebook posts, Preoţiuc-Pietro et al. [16] followed the same emotion model of Russell [14]. To predict these emotions they designed a bag-of-word prediction model using two linear regression models with $L2$ regularization (i.e., a technique used to avoid overfitting of the model).

There are other works which used word classification lexicons in order to predict the score of a given text [29]. In this work, the models consisted of a bag-of-words model (using TF and TF-IDF). The used lexicons were the three variants of the ANEW dataset (i.e., extended, bootstrapped and Warriner et al. [30] versions) and the evaluation was done in the ANET dataset [18]. In a first experiment, the lexicon which presented the best correlations was the Warriner et al. [30] dataset. They also extracted the RMSE metric, where curiously the lexicons that previously obtained the best correlations were the ones which presented the greatest errors. To improve these results, they trained linear regression models in the ANET dataset and passed as input the previously predicted values.

Later on, the same authors, applied k-NN regression models to create a mapping between the two main emotion models (i.e., dimensional and discrete) [31]. For their experiments they used 10-fold cross validation and a grid-search for the range [1, 100] in order to obtain the best k. They evaluated how their models generalize over different languages (i.e., English and Spanish) and datasets. They also tested their model over EmoBank dataset [17].

Another work by Kratzwald et al. [10] already used deep learning models to tackle the task of emotion detection in the circumplex model and in several categories. Their model consists of a layer of embeddings (i.e., either pre-trained GloVe or randomly-initialized trainable embeddings), followed by a recurrent layer (e.g., LSTM or BiLSTM), then a layer of dropout, and finally there is a layer responsible for forecasting the output (that can either be a category or a continuous numerical score). An interesting point in this work is the use of a technique called Transfer Learning, that consists of using other datasets (different from the one used for the problem) which are in some way related to the problem in question. Initially, the models are trained with these datasets, and the weights are then transferred to the new model (i.e., with the correct output layer), expecting an increase in performance. The results show that this model was able to outperform classic machine learning approaches, such as random forests, support vector machines, and bag-of-words approaches in several datasets.

One recent work joined similar tasks of emotion detection to create a model capable of learning combined representations [11]. This work focused on predicting fine-grained and coarse-grained emotions using deep learning methods. For this, they used CNN, LSTM and GRU architectures alongside a set of hand-crafted features (e.g., TF-IDF, lexicon, and Vader sentiment extraction tool) with GloVe embeddings. They first trained each model independently in a multi-task scenario. For each model, they extracted an intermediate layer which was concatenated together with the previously created hand-crafted feature vector. To handle every task, these hidden layers go through a Multi-layer perceptron, before forecasting the output for a given task. With this model, they took advantage of similar, but independent tasks to learn combined representations. The obtained results attest to the improvement of the use of a combined model over multiple single models.

3 The Proposed Approach

This work aims to use deep learning architectures in order to address the tasks of extraction and attribution of quotations, and classification of text regarding emotional valence and arousal dimensions. For this, it makes use of word embeddings to represent textual information which is explained in the first section. The following sections introduce the designed models for each of the tasks. These models have similar architectures, making use of different RNNs (e.g., LSTM, GRU, and Nested-LSTM) and multiple summarization layers (e.g., Attention and Max-pooling layers). Using distinct compositions of layers, there is the possibility of testing which combination achieved the best results in each task.

3.1 Representing Textual Information

Each token is represented as a 300-dimensional vector using the GloVe vectors which were trained in the Common Crawl[1] data using 840B tokens. To find out-of-vocabulary tokens, the system uses the Jaro-Winkler's similarity metric [32] to calculate the average between the vectors of the 2 closest tokens.

3.2 Models

All models (i.e., extraction, attribution, and classification models) were designed using Python[2] as the main programming language, alongside packages like *keras.io*[3] deep learning library with both Tensorflow and Theano backends.

1. http://commoncrawl.org/
2. http://www.python.org/
3. http://www.keras.io/

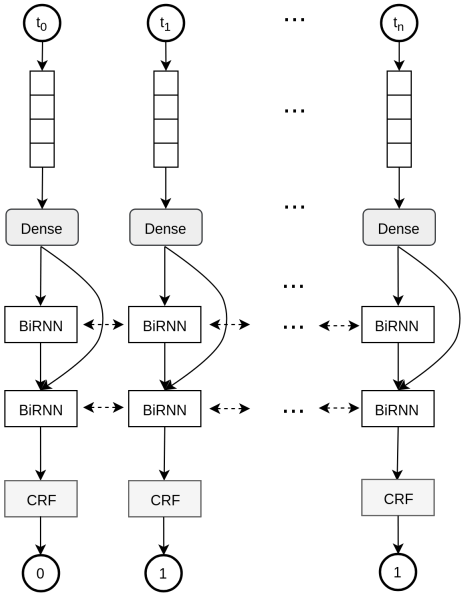

Fig. 1: Graphical representation of the quotation extraction model architecture.

For training, all models use Amsgrad optimizer [33] (i.e., variant of Adam optimizer) with the default *keras* hyperparameters. To prevent over-fitting of data, all models use a dropout layer (with a rate of 0.2) before the layer responsible for forecasting the prediction. I opted to train each model over a fixed-size number of epochs (10 epochs). This way the variants of each model could be fairly compared because there are no architectures being trained for longer than others.

3.2.1 Quotation Extraction

With this work, the goal was to demonstrate that by applying deep learning techniques it would not require any linguistic experience from the programmer. For this, the only pre-processing step that was done was the normalization of quotations marks to a single character token. No other type of normalization was executed, like lemmatization, lower-casing, or anonymization of entities or quotations. There is also no use of hand-crafted features.

Initially, the extraction model would receive each sentence as input, however, the way the dataset was annotated allows the existence of multi-sentence quotations. Not treating this problem could contaminate the results, i.e. quotations would be divided into several and would consequently consider quotations with incorrect types. For this, when parsing the dataset, the system verifies if a quotation still continues in the next sentence and if this happens it concatenates both sentences.

The complete model is represented in Figure 1 and it receives sentences that have been previously encoded to word embeddings. After that, there is a dense layer with a ReLU activation function, and there are four possibilities: (i) use a single bidirectional recurrent layer, (ii) use two stacks of BiRNN, (iii) use two stacks, with a shortcut between the dense layer and the last BiRNN layer; (iv) use a Nested-LSTM layer [19]. This last layer focus on nesting as an alternative to multi-layer stacking. It makes use of one inner and one outer LSTM cells, keeping a memory between both, creating a temporal hierarchy.

Finally, there is a CRF layer that classifies tokens as belonging (i.e., classifying as 1) or not (i.e., classifying as 0) to a quotation. For the training, the model uses the default loss function of the CRF layer.

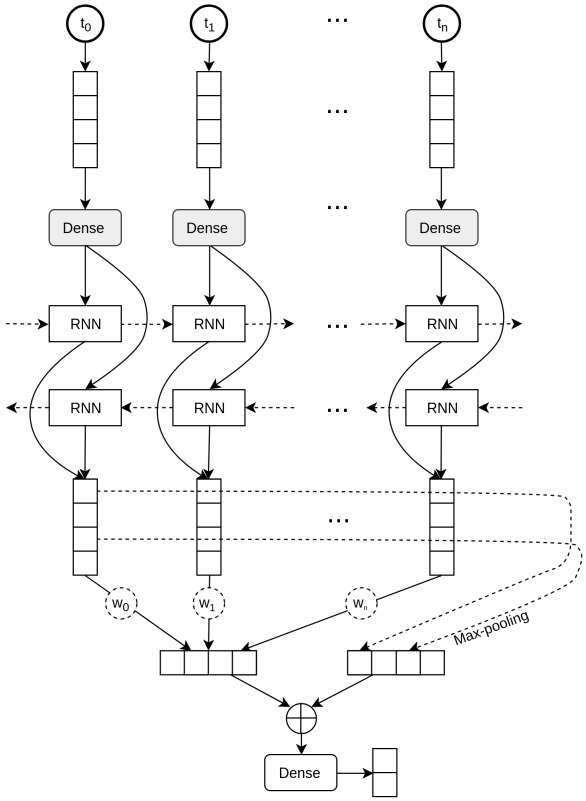

Fig. 2: Graphical representation of the architecture used in quotation attribution and sentence classification tasks.

3.2.2 Quotation Attribution

For the quotation attribution task, given a gold quotation and a set of possible candidates, the model needs to correctly attribute the correct candidate to the gold quotation. For this, I initially need to generate candidates. To do this, the tokens that are annotated in the dataset as Subject Noun Phrase (NP-SBJ) or Noun Phrase (NP) are used. Given this set of candidates, the system chooses the 3 closest candidates to the gold quotation (either ahead or behind) and that are within a maximum distance of 40 tokens. To these candidates, the true quotation speaker is added (if within the maximum distance and not inside the quotation span). The final goal of the model is to correctly disambiguate between a false candidate and the true speaker.

Figure 2 represents this architecture. For each entity, the model receives as input the span of text that goes from the candidate to the gold quotation. This span of text is then encoded using the same type of embeddings as the previous model (i.e., GloVe with Jaro-Winkler's distance). After that, it passes through a dense layer with a ReLU activation function. To capture the relations between words in a sentence, the model uses a bidirectional layer of recurrent neural networks (e.g., LSTM, GRU, and Nested-LSTM). After this, there is the choice of using a Max Pooling layer, an Attention layer, the concatenation of both, or not using any of the previous choices. If none of the previous was used, the last sequence from the bidirectional layer is returned. The final vector passes through a dense layer with a softmax activation function that

forecasts the probability of that candidate is the true source of the quotation (i.e., the complete span of text that represents the entity). The 2-dimension output vector represented in the Figure is due to the fact that this model uses categorical cross-entropy as the loss function. Meaning that the vector has both the probability of the candidate being the true speaker or not (summing to 1).

3.2.3 Emotion Classification

The complete sentence classification model is represented in Figure 2 and it is very similar to the quotation attribution model. It receives sentences as input, which were tokenized using *nltk* toolkit. Since punctuation plays a key role when predicting arousal values, e.g. exclamation marks (!) and ellipses (...), I decided to keep it.

The following layers (i.e., Dense with ReLU activation function, BiRNN, Attention, and Max Pooling) are the same as the attribution task and the final vector is then passed through a dense layer with a sigmoid activation function responsible for forecasting the values of valence and arousal.

Prior to the training of the model, there is a pre-processing step to normalize the values of valence and arousal to [0, 1] range. This model uses Mean Squared Error as the loss function. Before evaluating the results, there is another normalization step to put the scores into the original scale of the dataset.

4 Experimental Evaluation

In this section, I present all the process used to evaluate the designed models. The first section presents the datasets that were used for each of the proposed tasks. The next section introduces the metrics that were extracted to fairly compare this study with previous work. The last section describes all the conducted experiments throughout this paper on the different tasks and comparison with previous work.

4.1 Datasets

For the extraction and attribution of quotations, the models use PARC3 dataset [15], which contains about 20k retrieved and annotated attribution relations. Its annotation consists on defining the constituents of an attribution relation (i.e., content, source, and cue) and the properties of each token (e.g., lemma and POS tag). Additionally, it also contains the annotation for nested quotations (i.e., attribution relations inside others), however, this will not be part of this study. The original dataset is already pre-divided into train, test, and development folders and both models will respect this organization.

For evaluating the emotion detection task, the following datasets were used:

- Emobank [17, 34] is a dataset consisting of 10 thousand sentences that were manually annotated according to values of valence and arousal that were perceptible not only by the reader but also by the writer of the sentence. The ranges of values go from 1 to 5 points.
- Dataset encoding valence and arousal on Facebook posts [16], which contains 2895 Facebook posts ranked according to valence and arousal dimensions on a 9-point scale.
- The Affective Norms for English Text (ANET) dataset [18] contains 120 English sentences regarding its rating of emotion (e.g., pleasure, arousal, dominance) given by the mean value in a 9-point scale. Given its size, it will not be used as a training dataset.

4.2 Evaluation Metrics

To correctly evaluate the first task (i.e., quotation extraction), I will use the same evaluation metrics as previous works which consider strict and partial matches [1, 28]. For the strict measure, the predicted value is only correct if it fully matches the gold standard. The partial measure accounts for the proportion of correct overlapping tokens. The precision and the recall, for this metric, can be defined as follows:

$$\mathrm{P} = \frac{\sum_{g \in gold} \sum_{p \in pred} \mathrm{overlap}(g, p)}{|pred|} \tag{9}$$

$$\mathrm{R} = \frac{\sum_{g \in gold} \sum_{p \in pred} \mathrm{overlap}(p, g)}{|gold|} \tag{10}$$

In the previous equations p refers to the predicted quotation span, g the gold standard span, and $\mathrm{overlap}(x, y)$ is the proportion of tokens in y that are overlapped by x. This can be defined as follows:

$$\mathrm{overlap}(x, y) = \frac{|x \cap y|}{|y|} \tag{11}$$

As in previous works, individual metrics for each of the quotation type (i.e., direct, indirect and mixed) are also extracted.

For the quotation attribution task, I will apply the same strict measure, where the speaker is set to correct if it completely matches the gold standard, otherwise is marked as incorrect. For this task, it only makes sense to calculate the accuracy of the system, that is given by the correct predictions divided by the total number of predictions.

The last task (i.e., emotion detection) is evaluated using Mean Absolute Error (MAE) and Mean Squared Error (MSE) metrics for each of the dimensions. To better evaluate this task with previous work, Pearson correlation, r, which measures the degree of correlation between two variables, will also be used and is defined as follows:

$$\frac{\mathrm{covariance}(g, p)}{\sqrt{\mathrm{var}(g) \cdot \mathrm{var}(p)}} = \frac{\sum_{i=1}^{n} (g_i - \bar{g})(p_i - \bar{p})}{\sqrt{\sum_{i=1}^{n} (g_i - \bar{g})^2} \cdot \sqrt{\sum_{i=1}^{n} (p_i - \bar{p})^2}} \tag{12}$$

In the previous equation, g refers to the gold standard value, p the predicted value, and $\bar{g}$ and $\bar{p}$ refer to the mean value of the corresponding variable (i.e., valence or arousal dimensions).

4.3 Results

This section presents the obtained results according to the metrics defined in the previous section. For each of the tasks, multiple architectures variations were tested. This way there is the possibility of finding which combination brought the best results for each of the tasks. It starts by presenting the results regarding the quotation extraction task. The following section displays the results concerning the quotation attribution task. The last section provides the results obtained when classifying short texts in valence and arousal dimensions. Comparison with related work is done within each attention.

	Quotations											
	Direct			Mixed			Indirect			Overall		
Strict	P	R	F	P	R	F	P	R	F	P	R	F
Baseline	-	-	-	-	-	-	-	-	-	45.85	17.84	25.69
Bi-GRU	**94.35**	91.56	92.93	80.81	45.45	68.18	78.15	45.07	57.17	83.47	54.77	66.14
Bi-LSTM	92.46	**98.31**	**95.30**	79.33	**67.61**	73.01	79.24	**59.99**	**66.29**	82.85	**67.19**	**74.20**
2 * Bi-LSTM with shortcut	94.19	95.78	94.98	80.14	64.20	71.29	**83.12**	53.97	65.45	85.75	64.22	73.44
2 * Bi-LSTM without shortcut	93.85	96.62	95.22	**86.57**	65.91	**74.84**	82.80	52.74	64.44	**86.60**	63.87	73.51
Bi-NestedLSTM	92.62	95.36	93.97	75.59	54.55	63.37	80.37	42.05	55.22	83.53	55.03	66.35
Pareti [28]	94	88	91	67	60	63	78	56	65	80	63	71
Model Architectures [4]	93	94	94	81	66	73	73	65	69	79	71	75
Partial	P	R	F	P	R	F	P	R	F	P	R	F
Baseline	-	-	-	-	-	-	-	-	-	80.02	27.79	41.26
Bi-GRU	96.90	92.64	94.72	96.54	57.36	71.96	88.93	52.67	66.16	92.38	61.68	73.97
Bi-LSTM	96.24	**99.58**	**97.88**	**96.55**	**86.28**	**91.13**	87.61	**67.44**	**76.22**	91.40	**77.01**	**83.59**
2 * Bi-LSTM with shortcut	96.82	98.31	97.56	95.64	78.63	86.30	**92.37**	62.84	74.79	94.16	72.62	82.00
2 * Bi-LSTM without shortcut	**97.01**	98.44	97.72	96.17	77.11	85.59	90.88	61.54	73.38	93.50	71.59	81.09
Bi-NestedLSTM	96.60	97.39	96.99	96.48	73.11	83.18	92.33	52.85	67.22	**94.42**	65.20	77.14
Pareti [28]	99	93	96	91	81	86	91	66	77	93	73	82
Model Architectures [4]	97	95	96	92	81	86	83	75	79	88	80	84

TABLE 1: Performance of alternative models and related work when extracting quotations.

4.3.1 Quotation Extraction

This experiment had the goal of testing the quotation extraction model performance in PARC3 dataset. For this, I started by drawing a baseline that consists of recognizing the entire text between quotation marks as a quotation if its length is higher than 5 tokens. With this baseline, I intend to demonstrate not only the ease of recognizing direct quotations with a high precision, but also show that works which focus solely on detecting direct quotations are ignoring a large percentage of the total amount of quotations.

Table 1 presents the obtained results for the variants of my model. At first, the two types of RNN (i.e., GRU and LSTM) were tested. After this, some complexity was added, by using another bidirectional RNN stack with and without the shortcut step. The last line represents the experiment using the Nested-LSTM architecture as an alternative to layer-stacking. The table is divided into strict and partial metric results, reporting for each one the precision, recall and F-score. Finally, the previous work which focused on this task is also visible in the last lines of each metric.

Similarly to previous studies, single-token content spans have been ignored, removing them both from the set of gold and predicted spans [4, 28].

By analyzing the baseline results one can note that even though the precision is quite high, the low recall demonstrates that the other two types of quotations (i.e., indirect and mixed) should not be ignored. This baseline was designed to focus on direct quotations only, but by applying a partial measure it is also capable of extracting the direct type of mixed quotations (hence the high precision in the partial measure).

Overall, the LSTM architecture outperformed the GRU architecture. Looking at the lines which represent the use of two stacks of Bi-LSTM with and without shortcut connection, we can also see that adding more complexity did not bring an increase of performance over the use of a single Bi-LSTM. Additionally, using the Nested-LSTM layer, it did not bring the expected improvements over the use of multiple stacks. Concluding that, for this task, the architecture that presented the best results made use of a single bidirectional LSTM layer.

By the analysis of the table, one can also verify that it is crucial to remove metrics for all types of quotations since there is a discrepancy of results when recognizing the different types. The obtained results are in agreement with previous work by considering the indirect type the most difficult to extract.

Comparing with Pareti [28], my model generally obtained better results, being able to extract all types of quotations with an improvement of 3 points in the strict measure and 2 points in the partial measure.

When looking at the work by Scheible et al. [4], my model obtained very similar results: in the strict metric, it obtained 1 less point on the F-score for all the quotation types. The only type where my model was outperformed was over indirect type quotations. In the partial metric, my best model has generally obtained the same results as the compared work. It was, however, able to obtain better results in the first two types of quotations (i.e., direct and mixed) and only presented worse results when extracting indirect quotations.

4.3.2 Quotation Attribution

Similarly to the experiment regarding the extraction of quotations, in this experiment the goal is to test if the increase of complexity of the attribution model would improve the results. It was then tested for all types of RNNs (e.g., LSTM, GRU and Nested-LSTM) and with the option of using either a Max-pooling layer, an Attention layer, the concatenation of both or none of the previous.

For this, I conducted two different experiments: (i) One of the experiments (i.e., Conditionally Assigning Speakers) tests whether the model is capable of not only attributing the correct speaker but also not assigning a candidate when the quotation contains no annotated source. In this experiment, I considered all the quotations (even those without a source) and only returning the maximum probability candidate if it presents a probability greater than 0.5; (ii) Equally to Pareti [28], in a second experiment (i.e., Always Assigning Speakers), the quotations with no annotated source are ignored, and the system always returns the maximum probability candidate, i.e. even if it presents a probability lower than 0.5.

Table 2 shows the obtained results, reporting for each type of quotation the obtained accuracy. The presented baseline consists of a simple heuristic which assigns each gold quotation to the candidate that is closest to the beginning or the end of the quotation (also inserting the true speaker in the candidates set).

As can be seen from Table 2, the results were very positive, obtaining 88.28% of accuracy when quotations without speaker are considered, and 95.75% when these are ignored.

From the table, we can also conclude that: (i) attributing quotations to the nearest candidate, as represented in the

	Conditionally Assigning Speakers				Always Assigning Speakers			
	Direct	Mixed	Indirect	Overall	Direct	Mixed	Indirect	Overall
Baseline	38.82	36.16	44.69	42.23	39.32	38.32	51.75	46.89
Bi-GRU	91.98	88.70	84.27	86.49	96.58	93.41	94.69	94.91
Bi-LSTM	**95.78**	88.70	**85.85**	**88.28**	**98.29**	92.81	94.39	95.00
Bi-LSTM + Max Pooling (MP)	93.67	**91.53**	85.71	88.19	97.86	**97.60**	94.54	**95.75**
Bi-LSTM + Attention	92.41	87.01	**85.85**	87.34	96.15	94.61	94.54	94.91
Bi-LSTM + MP + Attention	93.25	86.44	84.14	86.32	97.44	94.61	**95.14**	95.57
Bi-NestedLSTM + MP	93.25	90.40	85.06	87.51	96.15	95.81	**95.14**	95.47
Pareti [28]	-	-	-	-	98	97	90	92

TABLE 2: Performance of alternative models and related work when attributing quotations.

baseline, is not enough to obtain good results; (ii) the LSTM architecture got slightly better results than the GRU; (iii) overall, the model which used a single Max-pooling layer was the one that brought the most significant improvements; (iv) the use of an Attention Layer did not bring advantages over not using it; (v) the concatenation of both summarization layers also did not improved the results when compared to a single Max-pooling layer; (vi) the use of a Nested-LSTM layer only brought improvements over the use of a simple LSTM, when attributing indirect quotations in the Always Assigning Speakers experiment; (vii) it is evident the difficulties in working with the distinct types of quotations, being the indirect type the hardest one to handle.

The only result which could be comparable is the work by Pareti [28], where she obtained an overall accuracy of 92% in the task of entity attribution (against my 96% accuracy in all quotations). However, in my work, I am not differentiating the tasks of entity attribution and source identification as she does. In her work, she starts by attributing the quotation to the correct entity and in the next step, she extends the entity to the complete source span, while I explicitly use the gold-standard source as a candidate. Even though the related work also uses gold-standard features and candidates, it is not fair to draw any comparison, since both works handle the task of quotation attribution in distinct ways.

4.3.3 Emotion Classification

This experiment presents the obtained results when inferring valence and arousal scores from textual sentences. For this, I started by designing some simpler models that served as comparison baselines. One first greedy approach could be designing a model that would use the available word classification datasets to predict the emotion scores of a sentence. This way I first designed a non-machine learning baseline that simply averages the score of valences and arousals for each word in the sentence. The values of valence and arousal for sentences composed solely of out-of-vocabulary words are given by the global mean of valence and arousal of the sentences dataset. I also created two extra baselines that already use machine learning approaches, both based on Support Vector Machines: one that uses as encoding of the sentence a bag-of-words and the other encodes a sentence as the average of all the embeddings vectors of the words belonging to the sentence.

This experiment had the goal to measure the importance of each component, and if the performance of the model gained with the increase of complexity. In order to test this, new components were progressively added (e.g., Attention and Max-pooling layer). It should be noted that two types of recurrent neural networks (e.g., LSTM and GRU) were tested, but only the results for the RNN that presented the best results (in this case LSTM) are shown. Another experiment evaluated if the use of a Nested-LSTM layer would benefit the results when compared to a simple LSTM layer.

Table 3 shows the results regarding this experiment. Apart from the system results, it also shows the performance of related works that in the past have already focused on the task of sentence classification regarding valence and arousal dimensions. For each of the models, I show the results independent of the dataset used for training and testing (e.g., EmoBank on the first column and Facebook posts in the right). All alternative models represented in Table 3 use cross-validation with k equal to 10.

Looking at the Table, some conclusions can be drawn:

- Despite its simplicity, the first baseline (i.e., Average Word Baseline) is able to present decent results in predicting the values of valence. However, it fails considerably when predicting arousal values.
- Comparing both SVM baselines approach, it is noticeable the improvement when using word embeddings over a bag-of-words approach. These results support the importance of word embeddings for sentence classification models.
- Analyzing both architectures of recurrent neural networks used (i.e., GRU and LSTM), the LSTM architecture presented the best results in both datasets.
- Looking at the lines that represent the models composed of Attention and Max-pooling layer, we can verify that the Max-pooling layer outperformed the Attention layer in both datasets. Also, the concatenation of both layers (i.e., Attention and Max-pooling) did not bring great improves over the use of the single Max-pooling layer.
- The last line indicates the experiments related to the use of a Nested-LSTM layer. We can see that in the EmoBank dataset, it still improved the results in some of the metrics (i.e., MAE and MSE when ranking valence scores). Regarding the Facebook posts dataset, it significantly improved the results in almost all metrics.

To measure the results in the EmoBank dataset, I compared them against the obtained inter-annotator agreement when creating the dataset. Although they obtained better correlations, my model is capable of predicting the different dimensions with a lower error.

Compared to the work from Buechel and Hahn [31] (which used a subset of the EmoBank dataset), my model presents the worst correlation in the valence dimension, but a better one when predicting arousal values. Looking at the mean of both dimensions, my model also performed slightly worst (0.492 cf. 0.448).

Comparing the results regarding Facebook posts dataset, with the work of Preoţiuc-Pietro et al. [16] and Buechel and

	EmoBank						Facebook posts					
	Pearson (r)		MAE		MSE		Pearson (r)		MAE		MSE	
	V	A	V	A	V	A	V	A	V	A	V	A
Average Word Baseline	0.387	0.164	0.475	0.312	0.359	0.161	0.394	0.014	1.174	2.013	2.086	5.464
SVM	0.268	0.241	0.295	**0.243**	0.168	0.102	0.442	0.783	0.807	0.999	1.152	1.625
Average Embedding + SVM	0.422	0.261	0.329	0.296	0.185	0.144	0.568	0.845	0.894	0.884	1.312	1.274
Bi-GRU	0.504	0.305	0.288	0.262	0.145	0.114	0.694	0.918	0.692	0.637	0.804	0.689
Bi-LSTM	0.525	0.313	0.277	0.259	0.135	0.111	0.702	0.919	0.673	0.634	0.781	0.698
Bi-LSTM+Attention	0.535	0.335	0.274	0.253	0.134	0.107	0.713	0.918	0.666	0.654	0.771	0.739
Bi-LSTM+Max Pooling (MP)	**0.566**	**0.361**	0.265	0.245	0.124	**0.100**	**0.725**	0.926	0.657	0.619	0.739	0.658
Bi-LSTM+MP+Attention	0.553	0.348	0.268	0.251	0.127	0.104	**0.725**	0.925	0.659	0.613	0.743	0.650
Bi-NestedLSTM+MP	0.559	0.346	**0.260**	0.248	**0.123**	0.102	0.712	**0.931**	**0.650**	**0.577**	**0.734**	**0.583**
Inter-annotator agreement [17]	0.738	0.595	0.349	0.441	-	-	-	-	-	-	-	-
Flexible Mapping Scheme [31]	0.788	0.227	-	-	-	-	-	-	-	-	-	-
Facebook Posts [16]	-	-	-	-	-	-	0.650	0.850	-	-	-	-
Affective Computing [10]	-	-	-	-	-	-	-	-	-	-	0.901*	3.346*
Regression Problem [29]	-	-	-	-	-	-	0.700	0.650	-	-	-	-
Multi-task Framework [11]	0.635	0.375	-	-	-	-	0.727	0.355	-	-	-	-

TABLE 3: Performance of alternative models to inferring valence and arousal scores from textual sentences.

Hahn [29] the model was able to outperform the obtained results.

The work by Kratzwald et al. [10] also uses the same dataset from the previous works. However, their work involved some pre-processing steps that I did not execute, such as normalizing the value range of valence and arousal to 0-10 and removing the punctuation (which I decided to keep since it is crucial when predicting arousal scores). By reproducing a similar experiment with the normalization of the values to the same range, I obtained an MSE of 1.291 and 1.004 for valence and arousal dimensions. My model presents a higher MSE when predicting valence, however, given that this is a squared metric, the error is not that significant. Regarding the arousal dimension, my model predicts with a significantly lower error, however, it is not fair to draw any conclusion since both works handle punctuation in a distinct way.

Comparing with the work Akhtar et al. [11], in the EmoBank dataset, their system achieved slightly better correlations when compared to mine. When tested in the Facebook dataset, my model had a lower, yet minimal correlation in the valence dimension (0.727 cf. 0.725), however in the arousal dimension my model presented much higher results (0.931 cf. 0.355).

In this experiment, I showed the performance of my model when trained and evaluated in different datasets (i.e., Facebook posts and EmoBank). In the EmoBank dataset, the performance of my model was significantly lower when compared to the obtained results in Facebook Posts dataset. However, this difference can be partly explained by the fact that both datasets have distinct annotation rules (by analyzing the related work, it is noted that EmoBank always presents lower errors and correlations). Regarding the dataset of Facebook posts, my model obtained very positive results, surpassing almost all the works that used the same dataset. Nevertheless, it would be interesting to observe in all related works, some metrics to measure the error (e.g., MAE or MSE). This would allow a fairer comparison of results.

There was another experiment that consisted of using the sentences datasets (i.e., EmoBank and Facebook posts) for training, and evaluate their performance in a smaller and unknown dataset (i.e., ANET). In this experiment, the larger dataset, i.e. EmoBank, obtained the best results obtaining correlations of 0.631 and 0.548 for valence and arousal respectively, and an MSE of 2.184 and 1.823 for the same dimensions. In the work of Buechel and Hahn [29] they obtained correlations of 0.710 and 0.640 for the dimensions of valence and arousal, and an MSE of 3.497 and 1.742, respectively. Comparing with my work, even though they obtained better correlation my model was able to present lower errors. Note that this work normalized the values for the range of -4 and 4, hence the discrepancy of values with previous experiments.

5 Conclusions and Future Work

Throughout this paper, I showed how tasks involving extraction and attribution of quotations, and classification of emotions in textual information can benefit from the use of neural networks architectures.

Initially, I built a model that upon receiving a newspaper article was capable of extracting all types of quotations (i.e., direct, indirect and mixed quotations). To the best of my knowledge, this is the first work which used deep learning in order to extract quotation spans. The model was tested over PARC3 dataset where it achieved results very close to the state-of-the-art. This confirms the advantages of these architectures over rule-based approaches that require domain expertise.

The second contribution of this work is a model designed to address the attribution of quotations in newspaper articles. In this task, my approach was slightly different from the often used in the literature: initially, I extracted the three closest candidates (to this set, the true speaker was added), and for each one, the text encompassing the candidate and the quotation was used as the input of a neural network. The candidate which presented the highest probability is set as the speaker. The final goal is to correctly disambiguate between the candidates and the true speaker of the quotation. In the same dataset as the previous task, this model was capable of correcly assigning the correct speaker with a 96% accuracy.

Having the possibility of extracting quotations and attributing them to the correct speaker, the missing component is a model capable of assigning these quotations to numerical scores of emotional valence and arousal dimensions. For this, an architecture was designed to classify sentences/short texts in Russel's circumplex model of emotion. The results were very positive, being able to overcome some of the works which focused on the same task.

The last contribution of this work is a publicly available website[4] where users can informally evaluate the complete pipeline (i.e., extraction, attribution, and classification of quotations).

4. http://emw.duckdns.org/~quotations/

Regarding future work, one of the ideas that could be tried out in all tasks would be to experiment new state-of-the-art representations. One possible choice for this could go through the use of ELMo[5] embeddings.

Another interesting idea to extend this project would be to try out recently proposed architectures such as the Transformer architecture [35] or the Capsule networks [36] which have been outperforming RNN and CNN architectures in some tasks.

For the task of emotion detection, capitalization of the words presents a key role when predicting emotional scores (especially arousal dimension). Depending on the training of the embeddings, these vectors may not be able to represent this type of information. An idea could be to use a set of handcrafted features to represent letter casing.

REFERENCES

[1] T. W. O'Keefe, "Extracting and attributing quotes in text and assessing them as opinions," Ph.D. dissertation, University of Sydney, 2014.

[2] M. S. C. Almeida, M. B. Almeida, and A. F. T. Martins, "A joint model for quotation attribution and coreference resolution," in *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, 2014.

[3] D. Jurafsky, A. X. Chang, G. Muzny, and M. Fang, "A two-stage sieve approach for quote attribution," in *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, 2017.

[4] C. Scheible, R. Klinger, and S. Pado, "Model architectures for quotation detection," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2016.

[5] B. Pouliquen, R. Steinberger, and C. Best, "Automatic detection of quotations in multilingual news," in *Proceedings of the International Conference Recent Advances in Natural Language Processing*, 2007.

[6] J. Liang, N. Dhillon, and K. Koperski, "A large-scale system for annotating and querying quotations in news feeds," in *Proceedings of the International Semantic Search Workshop*, 2010.

[7] A. Balahur, R. Steinberger, E. v. d. Goot, B. Pouliquen, and M. Kabadjov, "Opinion mining on newspaper quotations," in *Proceedings of the IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, 2009.

[8] T. O'Keefe, J. R. Curran, P. Ashwell, and I. Koprinska, "An annotated corpus of quoted opinions in news articles," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2013.

[9] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[10] B. Kratzwald, S. Ilic, M. Kraus, S. Feuerriegel, and H. Prendinger, "Decision support with text-based emotion recognition: Deep learning for affective computing," *arXiv preprint arXiv:1803.06397*, 2018.

[11] M. S. Akhtar, D. Ghosal, A. Ekbal, and P. Bhattacharyya, "A multi-task ensemble framework for emotion, sentiment and intensity prediction," *arXiv preprint arXiv:1808.01216*, 2018.

[12] M. Abdul-Mageed and L. Ungar, "EmoNet: Fine-grained emotion detection with gated recurrent neural networks," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2017.

[13] J. Godinho, B. Martins, and H. Cardoso, "Deep learning methods for predicting emotional valence and arousal in short texts," *(currently under revision)*, 2018.

[14] J. Russell, "A circumplex model of affect," *Journal of personality and social psychology*, vol. 39, no. 6, 1980.

[15] S. Pareti, "PARC 3.0: A corpus of attribution relations." in *Proceedings of Language Resources and Evaluation Conference*, 2016.

[16] D. Preoţiuc-Pietro, H. A. Schwartz, G. Park, J. Eichstaedt, M. Kern, L. Ungar, and E. Shulman, "Modelling valence and arousal in Facebook posts," in *Proceedings of the Association for Computational Linguistics Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 2016.

[17] S. Buechel and U. Hahn, "EMOBANK: Studying the impact of annotation perspective and representation format on dimensional emotion analysis," in *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, 2017.

[18] M. M. Bradley and P. J. Lang, "Affective Norms for English Text (ANET): Affective ratings of text and instruction manual," The Center for Research in Psychophysiology, University of Florida, Tech. Rep. D-1, 2007.

5. http://allennlp.org/elmo

[19] J. R. A. Moniz and D. Krueger, "Nested LSTMs," *arXiv preprint arXiv:1801.10308*, 2018.

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of the International Conference on Learning Representations*, 2013.

[21] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2014.

[22] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[23] Y. Goldberg, "A primer on neural network models for natural language processing," *Journal of Artificial Intelligence Research*, vol. 57, no. 1, 2016.

[24] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning in natural language processing," *arXiv preprint arXiv:1807.10854*, 2018.

[25] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, 1997.

[26] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[27] S. Pareti, T. O'Keefe, I. Konstas, J. R. Curran, and I. Koprinska, "Automatically detecting and attributing indirect quotations." in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2013.

[28] S. Pareti, "Attribution: A computational approach," Ph.D. dissertation, The University of Edinburgh, 2015.

[29] S. Buechel and U. Hahn, "Emotion Analysis as a Regression Problem - Dimensional Models and Their Implications on Emotion Representation and Metrical Evaluation," in *Proceedings of the European Conference on Artificial Intelligence*, 2016.

[30] A. B. Warriner, V. Kuperman, and M. Brysbaert, "Norms of valence, arousal, and dominance for 13,915 English lemmas," *Behavior Research Methods*, vol. 45, no. 4, 2013.

[31] S. Buechel and U. Hahn, "A flexible mapping scheme for discrete and dimensional emotion representations: Evidence from textual stimuli," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, 2017.

[32] W. E. Winkler, "String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage," in *Proceedings of the Section on Survey Research*, 1990.

[33] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond," in *Proceedings of the International Conference on Learning Representations*, 2018.

[34] S. Buechel and U. Hahn, "Readers vs. writers vs. texts: Coping with different perspectives of text understanding in emotion annotation," in *Proceedings of the Linguistic Annotation Workshop*, 2017.

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the Annual Conference on Neural Information Processing Systems*, 2017.

[36] W. Zhao, J. Ye, M. Yang, Z. Lei, S. Zhang, and Z. Zhao, "Investigating capsule networks with dynamic routing for text classification," *arXiv preprint arXiv:1804.00538*, 2018.