UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE CIÊNCIAS
DEPARTAMENTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FELIPE ZSCHORNACK RODRIGUES SARAIVA

ASPECT TERM EXTRACTION VIA BLSTM-CRF WITH ATTENTION ON WORD DEPENDENCIES

FORTALEZA
2019

FELIPE ZSCHORNACK RODRIGUES SARAIVA

ASPECT TERM EXTRACTION VIA BLSTM-CRF WITH ATTENTION ON WORD DEPENDENCIES

Dissertação apresentada ao Curso de do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Orientador: Prof. Dr. José Antônio F. de Macêdo

FORTALEZA

2019

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S246a Saraiva, Felipe Zschornack Rodrigues.
Aspect Term Extraction via BLSTM-CRF with Attention on Word Dependencies / Felipe Zschornack Rodrigues Saraiva. – 2019.
54 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação, Fortaleza, 2019.
Orientação: Prof. Dr. José Antônio Fernandes de Macêdo.

1. Aspect term extraction. 2. Attention mechanism. 3. Word dependencies. I. Título.
CDD 005

FELIPE ZSCHORNACK RODRIGUES SARAIVA

ASPECT TERM EXTRACTION VIA BLSTM-CRF WITH ATTENTION ON WORD DEPENDENCIES

Dissertação apresentada ao Curso de do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Aprovada em: 27 de Novembro de 2019

BANCA EXAMINADORA

Prof. Dr. José Antônio F. de Macêdo (Orientador)
Universidade Federal do Ceará (UFC)

Profa. Dra. Vládia Célia Monteiro Pinheiro
Universidade de Fortaleza (Unifor)

Prof. Dr. João Paulo Pordeus Gomes
Universidade Federal do Ceará (UFC)

Profa. Dra. Ticiana Linhares Coelho da Silva
Universidade Federal do Ceará (UFC)

Aos meus pais, meu irmão e meus amigos que, com muito carinho e apoio, não mediram esforços para que eu chegasse até esta etapa de minha vida.

AGRADECIMENTOS

Agradeço ao meu orientador, Prof. Dr. José Antônio Fernandes de Macêdo, por ter me orientado, ajudado e confiado na minha capacidade durante todo o desenvolvimento deste trabalho.

À Prof. Dr. Ticiana Linhares, por ter me auxiliado na escrita, sugerido melhorias no meu trabalho e por todo o apoio que me deu.

Aos meus pais, Soniamar Zschornack e João Santana, pelo amor, incentivo e apoio incondicional.

Ao Prof. Dr. João Paulo Pordeus Gomes e Prof.ª Dra Vládia Célia Monteiro Pinheiro por, generosamente, aceitarem meu convite e compor a banca.

À banca, pelo trabalho e empenho que tiveram em revisar este trabalho.

Ao meu amigo, Felipe Melo, sem o qual este trabalho provavelmente não existiria.

Aos meus amigos do Laboratório de Sistemas e Banco de Dados (LSBD), Eduardo Neto, Daniel Praciano, André Mendonça, Iago Chaves, Isabel Lima e Israel Vidal pelo apoio ao longo do desenvolvimento deste trabalho.

Aos meus colegas de trabalho, Judecir Cavalcante, Sávio Braga, Lana Mesquita, Tiago Malveira e Daniel Façanha por terem me incentivado ao longo deste trabalho.

À Universidade Federal do Ceará (UFC), em particular ao programa de Mestrado e Doutorado em Ciências da Computação, pela excelência de seus professores e servidores e pela oportunidade que me foi concedida.

A todos que direta ou indiretamente fizeram parte da minha formação.

Por fim, ao Doutorando em Engenharia Elétrica, Ednardo Moreira Rodrigues, e seu assistente, Alan Batista de Oliveira, aluno de graduação em Engenharia Elétrica, pela adequação do *template* utilizado neste trabalho para que o mesmo ficasse de acordo com as normas da biblioteca da Universidade Federal do Ceará (UFC).

“The good life is built with good relationships.”

(Robert J. Waldinger)

RESUMO

Neste trabalho propomos uma arquitetura de rede neural utilizando estruturas de aprendizado profundo, e engenharia de atributos mínima, para resolver o problema de extração de termos aspectos em textos opinativos, como avaliações de produtos e restaurantes. Em Mineração de Opiniões e Análise de Sentimentos existem três níveis de Classificação de Sentimentos: a nível de documento, a nível de sentença e a nível de aspecto. A Análise de Sentimentos Baseada em Aspectos visa encontrar o sentimento no nível de maior detalhe, e uma das tarefas que ela compreende é a Extração de Termos Aspectos (ETA): a identificação de aspectos (atributos ou características) que foram explicitamente avaliados dentro de uma sentença. Por exemplo, na sentença *"A qualidade das imagens dessa câmera é incrível."* o termo aspecto é *"qualidade das imagens"*. A arquitetura proposta consiste de um classificador BLSTM-CRF associado a um codificador BLSTM que utiliza um mecanismo de atenção para incorporar como atributo adicional no modelo as relações gramaticais entre as palavras (análise sintática) da sentença. Também utilizamos a marcação de partes da fala (part-of-speech tagging) como outro atributo relevante adicional. A arquitetura proposta obteve resultados semelhantes ao estado da arte, com a vantagem de não utilizar nenhuma regra de linguagem ou dicionário de sentimentos, apenas uma engenharia de atributos mínima.

Palavras-chave: Extração de termos aspectos. Mecanismo de atenção. Análise sintática. Memória de longo prazo bidirecional. Campos aleatórios condicionais

ABSTRACT

This work proposes a neural network architecture using deep learning structures, and minimal feature engineering, to solve the problem of aspect term extraction in opinionated documents, like reviews of products or restaurants. In Opinion Mining and Sentiment Analysis we have three levels of Sentiment Classification: document level, sentence level and aspect level. Aspect Based Sentiment Analysis aims to find the sentiment at the most detailed level, and one of its tasks is the Aspect Term Extraction (ATE): the task of identifying aspects (attributes or characteristics) that have been explicitly evaluated in a sentence. For example, in the sentence *"The picture quality of this camera is amazing"* the aspect term is *"picture quality"*. The proposed architecture consists of a BLSTM-CRF classifier with a BLSTM encoder that uses an attention mechanism to permit the incorporation of grammatical relations between words as an additional feature. We also used the Part-of-speech tag (POS tags) as another relevant feature. The proposed architecture obtained state-of-the-art results, with the advantage of using no linguistic rules or lexicons, only minimal feature engineering.

Keywords: Aspect term extraction. Encoder. Attention mechanism. Word dependencies. Bidirectional long short-term memory. Conditional random fields.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

With the advent of the internet and the social media, and their increasing usage, it is common to find people who share their opinions online about many things. Opinions are important specially when we need to make a decision, for example, when we want to buy something or decide what restaurant to go, we often read about the past experiences of other people about that product, place or company. But the shared opinions are not useful to customers only: companies can leverage the online opinions information to identify complains from their customers and improve their products and services.

Due to its importance to businesses and society, the field of *Sentiment Analysis and Opinion Mining* has become one of the most active research fields in text mining, with individuals and organizations trying to develop new data mining techniques to exploit this source of subjective information. By definition, the Opinion Mining consists of the automatic identification and extraction of opinions, emotions, and sentiments from text and multimedia data (LIU; ZHANG, 2012).

In Opinion Mining and Sentiment Analysis there are three levels of sentiment classification: document level, sentence level and aspect level. It is often imprecise to define the sentiment of an opinionated document or sentence, because the author can write both positive and negative opinions about different things in it. For example, imagine that we want to extract the sentiment of the following sentence:

"Nice ambience, but highly overrated place."

for that sentence, we can not precisely define if the overall sentiment is positive or negative, because both sentiments are present in it.

To address this problem, the Aspect-based Sentiment Analysis aims to find the sentiment at the most detailed level, the aspect level (LIU; ZHANG, 2012). One of the tasks involved in Aspect-based Sentiment Analysis is the aspect extraction: the identification of aspects (attributes or characteristics) that have been evaluated in an opinionated sentence.

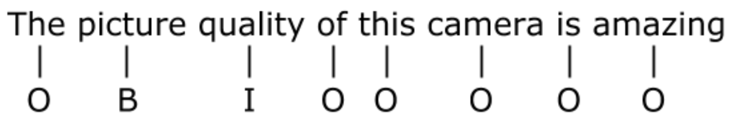
The Aspect Term Extraction (ATE) is the task of identifying the explicit aspects of a document or a sentence, i.e. the words in the document or sentence that represent an aspect. For example, in the sentence:

“The picture quality of this camera is amazing”

the aspect term is *“‘picture quality”*.

The Aspect Term Extraction task can be modeled as a classification problem, where each word in the sentence is tagged with a label. In this work, we use the IOB2 format (short for Inside, Out and Begin): words that are aspect terms are labelled with “B”; in case an aspect term consists of multiple words, the first word receives the “B” label and the remaining aspect words receive the “I” label; words that are not aspect terms are labelled with “O”. The Figure 1 illustrates the ATE task for the sentence *“The picture quality of this camera is amazing”*, using the IOB2 tagging format described.

Figure 1 – Aspect Term Extraction using the IOB2 tagging format.

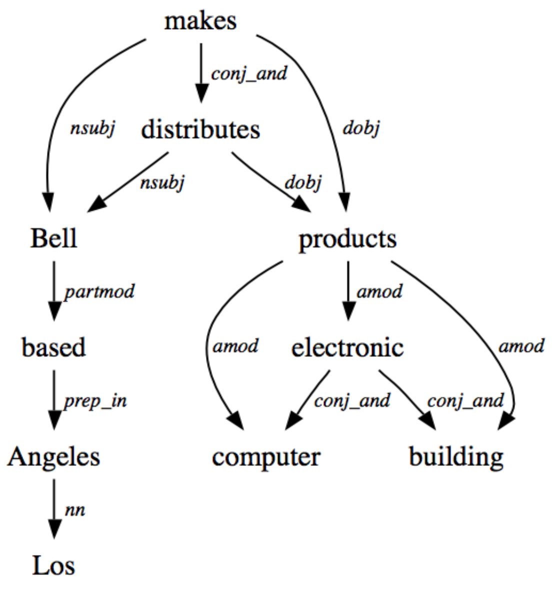


Different approaches, either supervised, unsupervised or semi-supervised, have been proposed to perform the task of aspect term extraction (RANA; CHEAH, 2016) and, more recently, the use of deep neural networks proved to be very efficient, becoming the state-of-the-art (PORIA *et al.*, 2016a). However, this technique requires a huge number of labeled examples to train the network (RAVI; LAROCHELLE, 2016), compelling some authors to compensate the low accuracy obtained from models trained on few examples with some kind of post-processing, auxiliary lexicons and language rules. Thus, in a scarce labeled data scenario, it is important to find relevant features that could improve the deep neural network models used for ATE, but without exaggerating on the manual feature engineering process. Manually engineered features, besides being demanding and costly to create, generally are problem-dependant and can restrict the final model to only one domain or dataset. It is also error prone, because a feature considered irrelevant by a human could actually be relevant to the problem and the model.

Aiming the use of deep neural networks to solve the ATE problem, but with minimal feature engineering to avoid its issues, this work proposes an architecture that achieves promising results in terms of performance using only two additional features: the Part-of-speech (POS) tag of words and the grammatical relationship between words in a sentence (word dependencies).

The novelty in our architecture is the use of an encoder, a common structure used in Neural Machine Translation (CHO *et al.*, 2014a; SUTSKEVER *et al.*, 2014), with an attention mechanism that allows the incorporation of the word dependency feature. The word dependency describes the grammatical relationship between the words, so it is a position related feature, where all words in the sentence can be associated with each other. Figure 2 shows an example of word dependencies for the sentence *"Bell, based in Los Angeles, makes and distributes electronic, computer and building products"*.

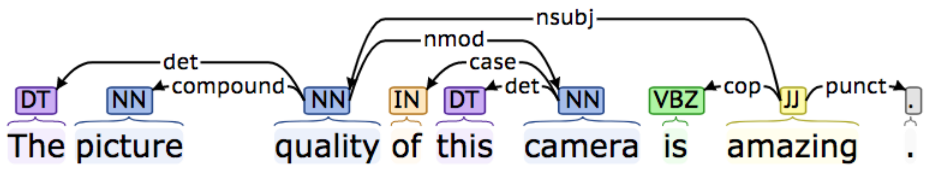
Figure 2 – Example of word dependencies generated using the Stanford Dependencies.



Due to this characteristic, it is not feasible to add this feature on the data by just concatenating it to the input words as we often do with other features (like the POS tag feature, or a binary feature indicating the case of the word's first letter). It is so, because the word dependency feature vector has the same size of the sentence and, as the sentences have different lengths, the input vectors would have different lengths also, which is not practical: all the input vectors of a neural network should have a fixed length. To overcome this problem, we could limit the size of the sentences to a maximum number of words, and pad the input vectors so they can all have the same length, but it is not desirable: our aim is to propose a model that can take as input sentences of any size. Another problem on concatenating the word dependency feature vectors on the input data is that it would be difficult for the model to identify the position information: what parts of the feature vector are associated with each word in the sentence, i.e. if there are 40 possible word dependencies, represented by one-hot encoded vectors, the model should figure out that the first 40 positions comprises the dependency vector related to the first word, the next 40 positions the dependency vector related to the second word, and so on.

The attention mechanism used on the encoder allows the incorporation of the word dependency feature, so the model can use the grammatical dependencies between the words to “pay attention” to certain parts of the sentence with relevant information, improving the identification of aspect terms. Figure 3 shows an example of word dependencies for the sentence *“The picture quality of this camera is amazing.”* generated by the Stanford Dependencies (MARNEFFE; MANNING, 2008).

Figure 3 – Example of word dependencies.



The word dependency feature is important to the problem of aspect extraction because aspect terms are usually nominal subjects (syntactic subjects) with an adjective associated. It also could help to identify multiple word aspect terms, using the compound dependency between nouns in a noun phrase. In this work we also use the Part-of-speech tag (POS tags) as another feature, because most of the aspect terms present in sentences are nouns associated with one or more adjectives (PORIA *et al.*, 2016a), thus the POS tagging information is useful to identify which words are aspect terms. Another reason to use these two additional features is to mitigate the problem of out of vocabulary (OOV) words, when there is no pre-trained word embedding to use as input to the model. It is important to mention that, unlike (PORIA *et al.*, 2016a; WANG *et al.*, 2016), we did not filter a subset of those features, we used all the existent POS tags and word dependencies and let the model select those that are most relevant in the training phase, i.e. we aimed to perform minimal feature engineering.

1.1 Contributions

The contribution of this work is: an end to end deep neural network architecture comprising an encoder with an attention mechanism strategy to harness the information of word dependencies to address the problem of aspect term extraction that, along with a POS tag feature, achieves competitive state-of-the-art results.

We carried out experiments to validate our proposal using popular datasets from the SemEval[1] competition. We compare our proposal with the winners of the SemEval competitions and state-of-the-art models presented in (PORIA *et al.*, 2016a) and (LI *et al.*, 2018). The experiments showed that our proposed architecture achieved promising results in terms of performance, similar to other state-of-the-art models, but using only minimal feature engineering.

1.2 Organization

This work is organized as follows. Chapter 2 presents the concepts and techniques necessary to understand the ATE task and our proposed neural network architecture. In chapter 3 we discuss about the state-of-the-art approaches and the related work, analyzing the strengths and flaws of them. In chapter 4 we give a detailed explanation of our proposed architecture. Chapter 5 details the experiments performed to assess our proposed architecture and compares it with state-of-the-art methods. Chapter 6 concludes the work.

[1] https://aclweb.org/aclwiki/SemEval_Portal

2 THEORETICAL FOUNDATION

2.1 Overview

In this chapter we present the concepts and structures necessary to understand the problem of Aspect Term Extraction and our proposed neural network architecture. We first introduce the field of Sentiment Analysis and define the Aspect Term Extraction task. Then we explain each structure, model and techniques used to build our neural network: the Recurrent Neural Networks (RNNs) used to handle sequential data; Long Short-Term Memory (LSTM), an RNN improvement; Bidirectional LSTMs, used to build our encoder and classifier; the Attention Mechanism, used on the encoder to exploit the word dependencies information; Linear-Chain Conditional Random Fields (CRF), a sequence optimization method to improve our classifier labelling process; and the Part-of-speech (POS) tags and Word Dependency features used to improve our model.

2.2 Sentiment Analysis

Sentiment Analysis is a subfield of Natural Language Processing, aiming to extract structured information from unstructured natural language text. The kind of information it aims to extract are opinions, that implies or express positive or negative sentiments about something.

Definition 2.2.1 (Opinion)**.** An opinion is a tuple (g, s) of a target g and a sentiment s associated with the target, where g can be any entity or aspect of the entity about which a view or judgment has been expressed, and s is a positive, negative, or neutral sentiment, or a numeric rating score expressing the strength/intensity of the sentiment (e.g., 1 to 5 stars) (LIU; ZHANG, 2012).

For example, in the sentence

"The picture quality of this camera is amazing, but I think it is too heavy."

the entity is *"camera"*, its aspects are *"picture quality"* and *"weight"*, and the sentiments of the aspects are *positive* and *negative* respectively. We can notice that it is not possible to define the *entity* (or the whole sentence) sentiment exactly, because both positive and negative sentiments are being expressed.

2.2.1 Aspect-based Sentiment Analysis

The Aspect-based Sentiment Analysis aims to find the sentiment at the most detailed level, the aspect level (LIU; ZHANG, 2012). One of the tasks involved in Aspect-based Sentiment Analysis is the aspect extraction: the identification of aspects (attributes or characteristics) that have been evaluated in a sentence. There are two types of aspects in an opinionated document or sentence: explicit aspects and implicit aspects.

Definition 2.2.2 (Explicit aspects)**.** Explicit aspects appears explicitly in the document or sentence, i.e. the words that compose the aspect are present in the sentence. For example, in the sentence *"The picture quality of this camera is amazing"* the explicit aspect is *"picture quality"*.

Definition 2.2.3 (Implicit aspects)**.** Implicit aspects, otherwise, do not appear in the sentence or document. For example, in the sentence *"This cellphone works for days on a single charge."*, we can infer that the implicit aspect is *battery life*.

Definition 2.2.4 (Aspect Term Extraction (ATE))**.** The Aspect Term Extraction (ATE) is the task of identifying the explicit aspects of a document or a sentence, i.e. the words in the document or sentence that represent an aspect.

The Aspect Term Extraction task can be modeled as a classification problem, where each word in the sentence is tagged with a label. One possible tagging format that can be used is the IOB2 format.

Definition 2.2.5 (IOB2 tagging format for ATE)**.** Given a sentence s of size n with words $(w_1, w_2, \cdots, w_n)$, assign for each word w_i one of the labels:

- **B** (begin), if the word is as aspect term;
- **I** (inside), if it is a subsequent word in an aspect term with multiple words;
- **O** (out), if it is not, or part of, an aspect term.

2.3 Recurrent Neural Networks

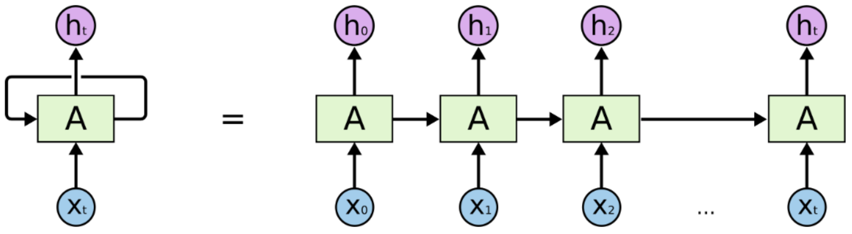
Recurrent Neural Networks (RNNs) emerged as a manner to treat sequential data using neural networks. Sequential data is any type of data where the order matters, i.e. the previous information is relevant, or can be used, to determine the current information. Examples of sequential data can be the words in sentences, the frames on movies, or the utterances in audio messages. It is not possible to guess the next word in a sentence without knowing the previous words, and it is difficult to know what is happening in the current frame of a movie without have seen the previous frames. As humans, our thoughts have persistence, and it would be useful if neural networks had the same ability and use the previous events in data to classify the current one. The RNN perform this using loops, allowing information in the network to be passed from one step to the next (ELMAN, 1990).

Let $(x_1, x_2, \cdots, x_n)$ be an input sequence of n elements. For each input x_t the RNN generates an output h_t, also called hidden output, that takes into account the input data x_t and the previous hidden output h_{t-1}:

$$h_t = a(W_h x_t + U_h h_{t-1} + b_t) \tag{2.1}$$

where a is an activation function, like tanh or the Rectified Linear Unit (ReLU), W_h and U_h are weight matrices, and b_t a bias vector. U_h is called transition matrix or recurrent matrix, and it determines how much of the past information is important on the current step. As the current hidden output h_t takes into account the last hidden output h_{t-1} and so on, all the hidden outputs contain traces of every preceded information. This information flux can be seen when we unroll a RNN, as shown in Figure 4.

Figure 4 – Unrolled RNN from (Christopher Olah, 2015).



2.4 Long Short-Term Memory (LSTM)

RNNs have problems to learn long sequences, i.e. in the presence of time lags greater than 5 - 10 discrete time steps, because as error signals exponentially depends on the magnitude of the weights (HOCHREITER, 1991), the backpropagated error quickly either vanishes, requiring an enormous amount of time to train the model, or blows up, leading to oscillating weights (HOCHREITER; SCHMIDHUBER, 1997; BENGIO *et al.*, 1994). This problem is known as the “vanishing/exploding gradient problem”.

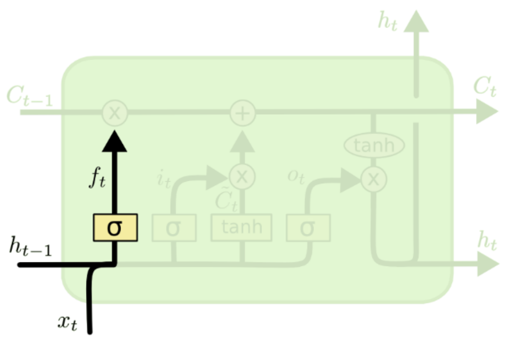
LSTMs are explicitly designed to avoid the long-term dependency problem, when there is a big gap between the present task to the previous information, or context, needed to solve it. For example, if we want to predict the next word in the text "I grew up in France… I speak fluent *French*", the context word France may be further back in the text, so the gap between the relevant information and the point where it is needed is very large.

While in a RNN each neuron or cell has only one structure, like a *tanh* activation function, in a LSTM cell unit there are four structures: the cell state and three gates. The LSTM cell state keeps and transfer information through the sequence chain, working like a memory. The cell state can carry relevant information throughout the sequence, reducing the effects of short-term memory.

The LSTM can control the cell state removing or adding information to it through minor linear interactions with gates. The gates are structures that control whether information go through the steps so, using the gates, it is even possible for the cell state information to flow along the steps unchanged. They are composed of a sigmoid neural net layer and a pointwise multiplication operation. The LSTM gates that control the cell state information are: the forget gate, the input gate and the output gate.

The **forget gate** decides what information should be kept. Information from the previous hidden state and information from the current input are passed through a sigmoid function. If the output of the sigmoid function is zero, no information goes through, if it is one, it passes all the information to the next step. Figure 5 demonstrates the forget gate operation.

Figure 5 – Illustration of forget gate operation from (Christopher Olah, 2015).



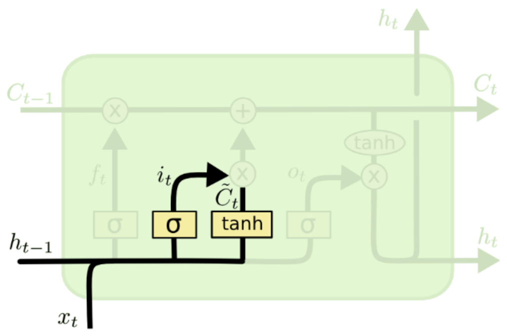
Definition 2.4.1 (Forget Gate)**.** Let n be the number of LSTM cell units, $f_t \in \mathrm{R}^n$ the forget gate's activation vector for state t, σ the sigmoid function, W_f the weight matrix of the input connections for the forget gate, U_f the weight matrix of the recurrent connections for the forget gate, x_t the input for the current state t, h_{t-1} the hidden outputs of the previous state and b_f a

bias vector, the forget gate's activation vector f_t for state t is computed by:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{2.2}$$

The **input gate** controls what new information is going to be stored in the cell state. It applies a sigmoid function to the current state input and the previous hidden state. The input gate vector is then multiplied by the new values to be stored in the cell state. Figure 6 demonstrates the input gate operation.

Figure 6 – Illustration of input gate operation from (Christopher Olah, 2015).

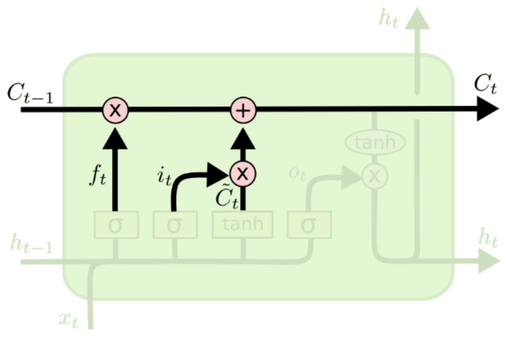


Definition 2.4.2 (Input Gate)**.** Let n be the number of LSTM cell units, $i_t \in \mathbb{R}^n$ the input gate's activation vector for state t, σ the sigmoid function, W_i the weight matrix of the input connections for the input gate, U_i the weight matrix of the recurrent connections for the input gate, x_t the input for the current state t, h_{t-1} the hidden outputs of the previous state and b_i a bias vector, the input gate's activation vector i_t for state t is computed by:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{2.3}$$

The **cell state** is computed using the forget gate vector and the input gate vector information. First, the previous cell state is element-wise multiplied by the forget vector. It is possible to drop values in the cell state if the values in the forget gate vector are near to zero. The current state input and the previous hidden state are passed through a *tanh* function, calculating the new candidate values to be stored in the cell state. These values are element-wise multiplied by the input vector, to decide what information is going to be stored in the cell state. The values obtained from the forget and input gates are then added (element-wise addition), generating the current cell state information. Figure 7 demonstrates the cell state computation.

Figure 7 – Illustration of cell state computation from (Christopher Olah, 2015). $\tilde{C}_t$ represents the new candidate values to be stored in the cell state.



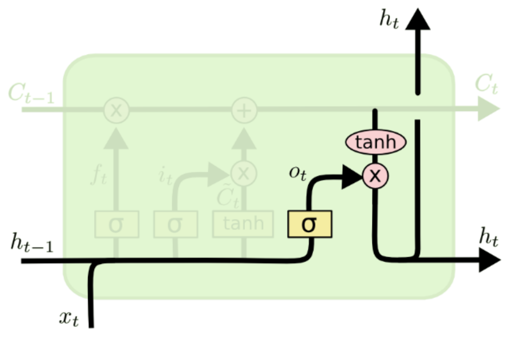
Definition 2.4.3 (Cell State)**.** Let n be the number of LSTM cell units, $c_t \in \mathbb{R}^n$ the cell state for state t, σ the sigmoid function, W_c the weight matrix of the input connections for the cell state, U_c the weight matrix of the recurrent connections for the cell state, x_t the input for the current state t, h_{t-1} the hidden outputs of the previous state and b_c a bias vector, the cell state vector c_t for state t is computed by:

$$c_t = f_t \circ c_{t-1} + i_t \circ \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{2.4}$$

where $\circ$ denotes the element-wise product.

Finally, the **output gate** decides what the next hidden state values should be. It pass the current state input and the previous hidden state through a sigmoid function, and the newly modified cell state through a tanh function. The results are then multiplied to generate the new hidden state. The new cell state and the new hidden are then carried over to the next time step. Figure 8 demonstrates the output gate operation.

Figure 8 – Illustration of output gate operation from (Christopher Olah, 2015).



Definition 2.4.4 (Output Gate)**.** Let n be the number of LSTM cell units, $o_t \in \mathbb{R}^n$ the output gate's activation vector for state t, σ the sigmoid function, W_o the weight matrix of the input connections for the output gate, U_o the weight matrix of the recurrent connections for the output gate, x_t the input for the current state t, h_{t-1} the hidden outputs of the previous state and b_o a bias vector, the output gate's activation vector o_t for state t is computed by:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{2.5}$$

Definition 2.4.5 (Hidden State)**.** Let n be the number of LSTM cell units, $h_t \in \mathbb{R}^n$ the hidden state's output vector for state t and c_t the cell state for state t, the hidden state's output vector h_t for state t is computed by:

$$h_t = o_t \circ \tanh(c_t) \tag{2.6}$$

where $\circ$ denotes the element-wise product.

2.5 Bidirectional LSTM (BLSTM)

In natural language it is beneficial to have access to future information. In a conversation, for example, a word that at first we didn't understand makes sense on the presence of future context (GRAVES; SCHMIDHUBER, 2005). In BLSTMs we give each sentence as input to one LSTM layer (forward) and the inverted sentence as input to another LSTM layer (backward) to capture both past and future information.

Let n be the number of steps in the input sequence, the forward LSTM $\overrightarrow{f}$ reads the input sequence as it is ordered (from x_1 to x_n) and calculates a sequence of forward hidden states $(\overrightarrow{h_1}, \cdots, \overrightarrow{h_n})$. The backward LSTM $\overleftarrow{f}$ reads the input sequence in the reverse order (from x_n to x_1) and calculates a sequence of backward hidden states $(\overleftarrow{h_1}, \cdots, \overleftarrow{h_n})$.

The final hidden state h_j for each word x_j is obtained concatenating the forward hidden state $\overrightarrow{h_j}$ and the backward hidden state $\overleftarrow{h_j}$.

$$h_j = [\overrightarrow{h_j}; \overleftarrow{h_j}] \tag{2.7}$$

2.6 Linear-Chain Conditional Random Field (CRF)

In a sequence labelling problem we want to find the sequence of labels y for a given input X. Instead of finding the labels independently, we could explore some existing dependencies between them. For example, in the ATE problem, an inside (“I”) tag can appear only after a begin (“B”) tag. A Linear-Chain Conditional Random Field (CRF) works as a sequence optimization model, that jointly decode the best label sequence. The CRF considers the correlations between the neighbours labels, making a global choice instead of decoding each label of the sequence independently. In the problem of ATE it is specifically useful to correctly label multiple word aspect terms (LAMPLE *et al.*, 2016).

Definition 2.6.1 (Linear-Chain Conditional Random Field (CRF))**.** Given an input sequence

$$X = (x_1, x_2, \cdots, x_n) \tag{2.8}$$

of size n, with label predictions

$$y' = (y'_1, y'_2, \cdots, y'_n), \tag{2.9}$$

we define the label predictions score s as:

$$s(X, y') = \sum_{i=1}^{n} T_{y'_i, y'_{i+1}} + \sum_{i=1}^{n} H_{x_i, y'_i} \tag{2.10}$$

where H and T are both matrices of scores. H is a matrix with size $n \times l$ where l is the number of distinct labels, and $H_{i,j}$ represents the score of the j^{th} label of the i^{th} element in the sequence (H can be, for example, the outputs of a BLSTM). T is a matrix with size $l \times l$ where $T_{i,j}$ represents the score of a transition from the label i to the label j. The transition scores $T_{i,j}$, as well as the scores $H_{i,j}$, are learnable.

A probability for a label sequence y' can be calculated using a softmax over all the possible label sequences:

$$p(y'|X) = \frac{e^{s(X,y')}}{\sum_{\tilde{y} \in Y_X} e^{s(X,\tilde{y})}} \tag{2.11}$$

where Y_X is the set of all possible label sequences.

In the training, we maximize the log-probability of the correct label sequence:

$$\log(p(y'|X)) = s(X,y') - \log(\sum_{\tilde{y}\in Y_X} e^{s(X,\tilde{y})}) \tag{2.12}$$

In the decoding phase, we return the output label sequence y^* with the maximum score:

$$y^* = \underset{\tilde{y}\in Y_X}{\operatorname{argmax}}\, s(X,\tilde{y}) \tag{2.13}$$

We can exploit the recurrent dependencies in Eq. 2.12 and Eq. 4.4 and use dynamic programming to compute them efficiently.

2.7 Attention Mechanism

The attention mechanism was originally used in the field of Neural Machine Translation to improve the performance of Encoder-Decoder neural architectures. It is often used in an encoder, allowing the model to automatically (soft-)search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly (BAHDANAU *et al.*, 2014), i.e. it does not attempt to encode the whole input sentence into a single fixed-length vector. It is specially useful for long sentences, but the improvements can be observed with sentences of any length. Instead of generating the same context vector, or sentence representation, for all words in the input sentence, the attention mechanism allows the encoder to compute a different context vector for each word based on an alignment model or function. This alignment model or function gives a score of how well the word and the encoder output match.

Definition 2.7.1 (Context Vector)**.** Given a sentence of size n with words $(w_1, w_2, \cdots, w_n)$, a sequence of outputs $(h_1, h_2, \cdots, h_n)$ to which the encoder maps the input sentence and a an alignment function, the context vector c_i for the i^{th} word is given by:

$$c_i = \sum_{j=1}^{n} \alpha_{i,j} h_j \tag{2.14}$$

The terms $\alpha_{i,j}$ are the attention weights of each output h_j. They are computed by:

$$\alpha_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k=1}^{n} \exp(e_{i,k})} \tag{2.15}$$

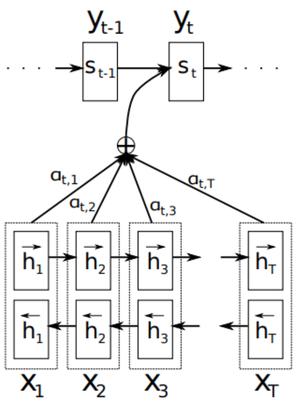
where

$$e_{i,j} = a(w_i, h_j) \tag{2.16}$$

are the scores of how well the word w_i and the output h_j match.

Hence, the context vector is a weighted sum of the outputs generated by the encoder, where the attention weights (denoted by $\alpha_{i,j}$) are computed using an alignment model or function. Figure 9 illustrate this process.

Figure 9 – Illustration of the attention mechanism from (BAHDANAU *et al.*, 2014).



A decoder or a classifier can then use the context vectors generated by the encoder with the attention mechanism to improve its performance, specially on long sentences.

2.8 Part-of-speech tags (POS tags)

A part-of-speech (POS) is a category of words. Words that are assigned to the same part of speech generally display similar syntactic behavior: they play similar roles within the grammatical structure of sentences (HASPELMATH, 2001). The ten common POS classes in English are:

1. noun
2. verb

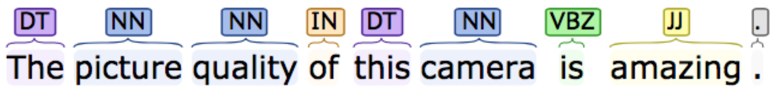
3. adjective
4. adverb
5. pronoun
6. preposition
7. conjunction
8. numeral
9. article
10. interjection

Other fine-grained POS classes can be derived through inflection, where the word is modified to express different grammatical categories like tense, gender or number.

The POS class or tag information of a token is very useful in the field of NLP, because the POS tag is assigned to a token based both on its definition and its context, so it also carries information about the relationship that token has with adjacent words in the sentence.

The POS tagging process is done using a POS tagger, like The Stanford POS Tagger (TOUTANOVA *et al.*, 2003). The Stanford POS Tagger uses a fine-grained set of tags: the Penn Treebank tag set (MARCUS *et al.*, 1993). Figure 10 shows an example of the POS tags for the sentence *"The picture quality of this camera is amazing."* generated by the Stanford POS Tagger.

Figure 10 – POS tags for the sentence *"The picture quality of this camera is amazing."*.



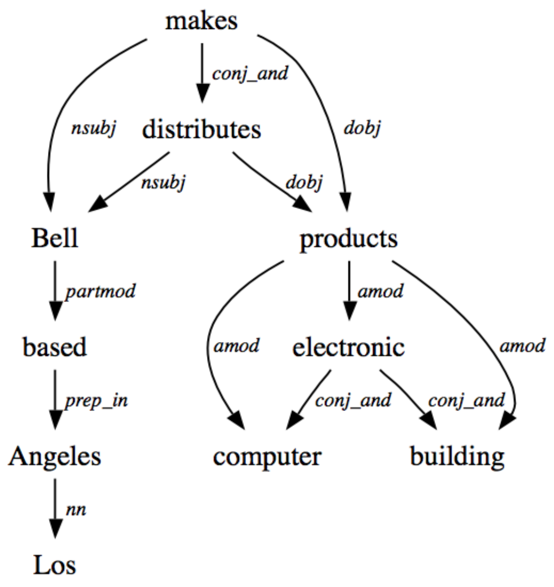
2.9 Word Dependencies

Word Dependencies are the descriptions of the grammatical relationships between words in a sentence (MARNEFFE; MANNING, 2008).

They are represented as a directed graph, where each node is a word in the sentence and the edges are the grammatical relations between them. Figure 11 shows an example of word dependencies graph for the sentence *"Bell, based in Los Angeles, makes and distributes electronic, computer and building products"* generated using the Stanford Dependencies (MARNEFFE; MANNING, 2008).

Differently from part-of-speech (POS) tags, the word dependencies represents all sentence relationships uniformly as typed dependency binary relations and are used in NLP to extract information about textual relationships. The typed relations have the form: *dependency_type*(dependant_word, governor_word), e.g. *nsubj*(makes, Bell).

Figure 11 – Graph of word dependencies for the sentence *"Bell, based in Los Angeles, makes and distributes electronic, computer and building products"* generated using the Stanford Dependencies.



3 RELATED WORK

Different approaches, either supervised, unsupervised or semi-supervised, have been proposed to perform the task of aspect term extraction (RANA; CHEAH, 2016) and, more recently, the use of deep neural networks proved to be very efficient, becoming the state-of-the-art in this field (PORIA *et al.*, 2016a).

3.1 Previous Approaches for ATE

The earlier approaches to perform Aspect Term Extraction were generally unsupervised and used metrics such as the frequency of some categories of words, like nouns and noun phrases to identify aspect terms. The assumption was that the words representing aspects occur more frequently in an opinionated text than other words, what is not true for some texts where aspects are infrequent terms (HU; LIU, 2004).

Another approach was to use rule based linguistic-patterns, to exploit the aspect and sentiment relation (DO *et al.*, 2019). In these works, the authors proposed a set of rules to first identify the sentiment word and then use grammatical relations to obtain the aspect word. The assumption was that the sentiment words are easier to find, generally using sentiment lexicons (dictionaries) and the word's category (adjective). Aspect lexicons could also be used to improve the results (PIRYANI *et al.*, 2017).

The topic modelling technique, normally using the Latent Dirichlet Allocation (LDA) algorithm, was also used to solve the ATE problem. The LDA algorithm identify different latent "topics" in the documents, where each document contains many topics, and each topic is represented by a set of relevant words. The topics found were considered as the aspects present in the documents and then were manually labeled (PORIA *et al.*, 2016b)

Supervised techniques have also been used to perform the ATE task. In the SEMEVAL 2014-2016 competitions, many models were built using Support Vector Machines (SVM) and Conditional Random Fields (CRF), with CRF models being the top performers (CHERNYSHEVICH, 2014; TOH; WANG, 2014; TOH; SU, 2016).

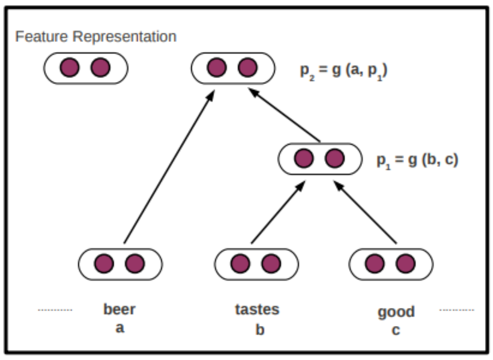
3.2 Deep learning models for ATE

Recently, some deep learning models have been proposed to solve the problem of aspect term extraction. Deep learning architectures have the ability to learn representations of data using multiple layers where each layer represents a level of abstraction (LECUN *et al.*, 2015). This characteristic, of performing automatic feature engineering, allow them to capture both syntactic and semantic features from the text (DO *et al.*, 2019).

3.2.1 Aspect Specific Sentiment Analysis using Hierarchical Deep Learning

Lakkaraju, Himabindu, Richard Socher, and Chris Manning (2014) (LAKKARAJU *et al.*, 2014) proposed a hierarchical deep learning structure to learn representations for words (embeddings) which aim to explain the aspect-sentiment relationship at the phrase level. Their model used the dependency parse of the phrase to compute the embeddings using an RNN, where each level of the tree were represented by an RNN embedding. The embeddings learned were then used to the joint modeling of aspects and sentiments, for posterior aspect and sentiment extraction. Figure 12 shows a scheme of the representations computed for each level of the parse tree. The higher levels of the tree uses the representations computed on the lower levels as input. Their model achieved state-of-the-art performance for 2014, the year when it was published.

Figure 12 – Example of RNN representations computed using the phrase's dependency parse tree.



3.2.2 Aspect extraction for opinion mining with a deep convolutional neural network

Poria, Cambria, and Gelbukh (PORIA *et al.*, 2016a) presented a 7-layer Convolutional Neural Network (CNN) with a CRF layer, along with a set of linguistic rules to tag each word in sentences as being aspects or not. The CNN used five-grams as inputs, i.e. the CNN inputs were a window of 5 words around each word in the sentence. In their work they made use of domain specific pre-trained embeddings as input features and also used a sentiment lexicon, and Part of Speech (POS) vectors as handcrafted features to improve their model performance. For the POS vectors, they manually selected 6 basic parts of speech (noun, verb, adjective, adverb, preposition, conjunction) encoding it as a 6-dimensional binary vector for each input word. Their model obtained the best state-of-the-art results so far.

3.2.3 Unsupervised Aspect Term Extraction with B-LSTM & CRF using Automatically Labelled Datasets

Giannakopoulos, Athanasios, et al. (GIANNAKOPOULOS *et al.*, 2017) proposed a two layer bidirectional LSTM & CRF model, trained on automatically labeled datasets to extract aspects from opinionated sentences, which configures a weak supervision approach. They created a rule based unsupervised algorithm to automatically label the datasets used to train their deep learning model. Their BLSTM-CRF model trained with and without the automatically labeled dataset presented a better performance when compared to the best models of the SemEval 2014 competition.

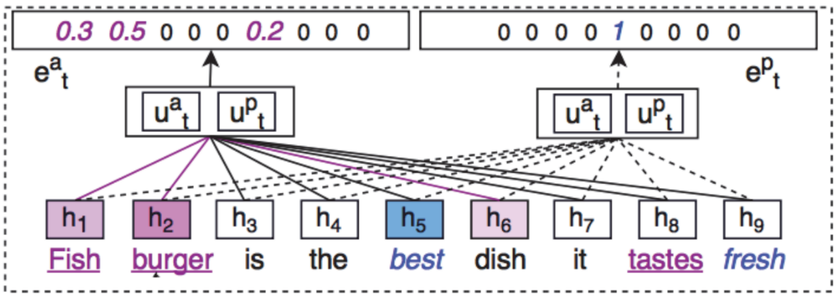
3.2.4 Recursive Neural Conditional Random Fields for Aspect-based Sentiment Analysis

Wang et al. (WANG *et al.*, 2016) improved the dependency-tree RNN (DT-RNN) proposed by (LAKKARAJU *et al.*, 2014) with a CRF layer and three hand-crafted features: POS tags, name-list and sentiment lexicon to perform the tasks of aspect term extraction and opinion term extraction at the same time. The motivation of using the DT-RNN to encode grammatical dependency between words for feature learning was because it is infeasible or difficult to incorporate dependency structure explicitly as input features. They achieved state-of-the-art results with their model.

3.2.5 Coupled Multi-Layer Attentions for Co-Extraction of Aspect and Opinion Terms

Wang et al. (WANG *et al.*, 2017) used an attention mechanism to identify both aspect and opinion terms in a sentence: one attention layer to identify aspect terms and other attention layer to identify opinion terms. Their goal was to avoid using engineered features, like word dependencies. In their work, each attention layer learns a prototype vector, a general feature representation for aspect terms and opinion terms. The attention weights measure the extent of correlation between each input token and the prototypes using a tensor operator. Tokens with high attention weight values are labeled as aspect or opinion terms. The attention layers were coupled to fully exploit the relations between aspect terms and opinion terms. Figure 13 shows a scheme of their proposed architecture. Their proposed model obtained state-of-the-art results.

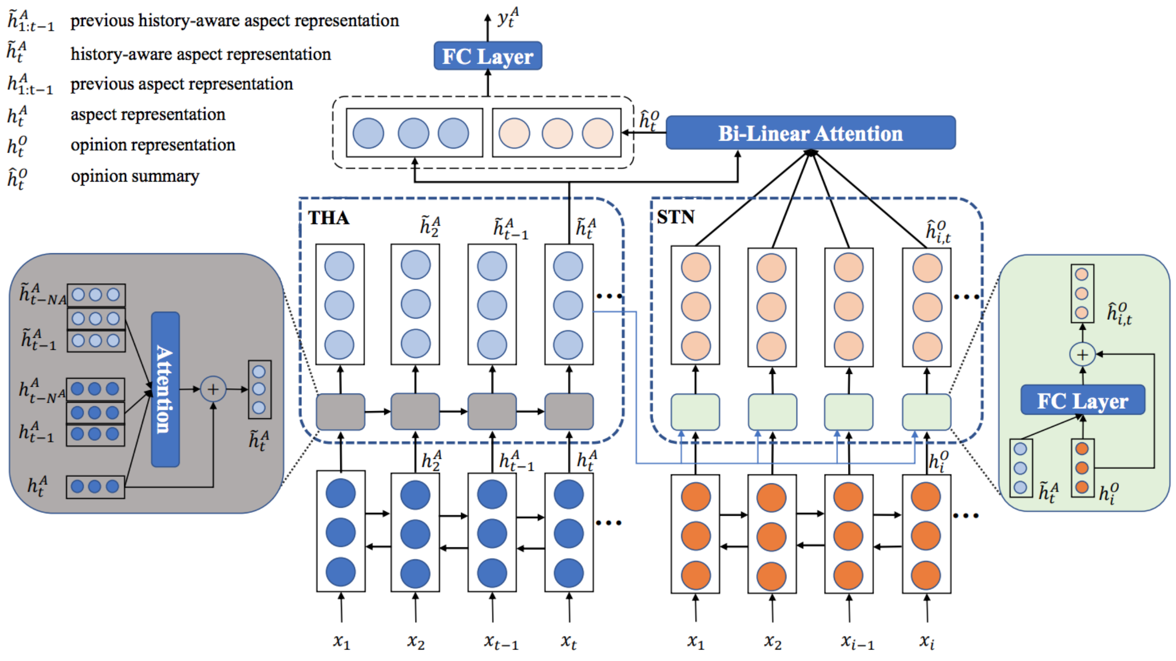
Figure 13 – Multi-layer coupled attentions architecture. u^a and u^p are the prototype vectors for aspects and opinions respectively.


0.3 0.5 0 0 0 0.2 0 0 0
0 0 0 0 1 0 0 0 0
e^a_t
e^p_t
u^a_t u^p_t
u^a_t u^p_t
h_1 h_2 h_3 h_4 h_5 h_6 h_7 h_8 h_9
Fish burger is the best dish it tastes fresh

3.2.6 *Aspect Term Extraction with History Attention and Selective Transformation*

Li, Xin, et al. (LI *et al.*, 2018) proposed a new framework to perform the ATE task using the opinion information. They proposed a component called Truncated History-Attention (THA) to exploit the relation between previous predictions and the current prediction, improving the detection of multiple word aspect terms and the IOB labelling process. The THA uses an attention mechanism that computes an history-aware aspect representation vector based on all the previous steps. For each step, it compares this history-aware representation with the current aspect representation and updates the history-aware aspect representation vector. It also proposes a Selective Transformation Network (STN) that uses the aspect representation from THA and an opinion summary vector to compute a specific opinion representation for each word in the sentence. After, a bi-linear attention mechanism is used to update the opinion summary vector as the weighted sum of the new opinion representations, according to their associations with the current aspect representation. They achieved the best state-of-the-art result so far for the SEMEVAL 2016 Restaurant domain dataset. Figure 14 shows their framework architecture.

Figure 14 – The framework architecture proposed by (LI *et al.*, 2018) using THA and STN for aspect extraction using the opinion information.


$\tilde{h}^A_{1:t-1}$ previous history-aware aspect representation
$\tilde{h}^A_t$ history-aware aspect representation
$h^A_{1:t-1}$ previous aspect representation
h^A_t aspect representation
h^O_t opinion representation
$\hat{h}^O_t$ opinion summary
y^A_t
FC Layer
Bi-Linear Attention
THA
STN
Attention
x_1
x_2
x_{t-1}
x_t
x_{i-1}
x_i

3.3 Discussion

Table 1 compares the main models used for ATE covered in this section. We can notice that the deep learning models have better performance when compared to the classical approaches. We can also observe that the deep learning models only, without additional features, are not able to obtain the best results in the ATE task, like the BLSTM-CRF proposed by (LAKKARAJU *et al.*, 2014). On the other hand, the best ATE model so far, proposed by (PORIA *et al.*, 2016a), used too many hand crafted features and rules, what is demanding and can be error prone. Thus, in a scarce labeled data scenario, it is important to find relevant features that could improve the deep neural network models used for ATE, but without exaggerating on the manual feature engineering process. Also, the recent works using attention mechanisms showed that this technique can be explored to create novel architectures that adds, and take advantage of, relevant information to solve the ATE problem, like the opinion information explored by the framework proposed in (LI *et al.*, 2018).

Table 1 – Overview of different approaches used to perform Aspect Term Extraction and their performance on the SemEval 2014 datasets. Classical approaches that do not use Deep Learning are marked with (#). L stands for the Laptop dataset and R for the Restaurant dataset.

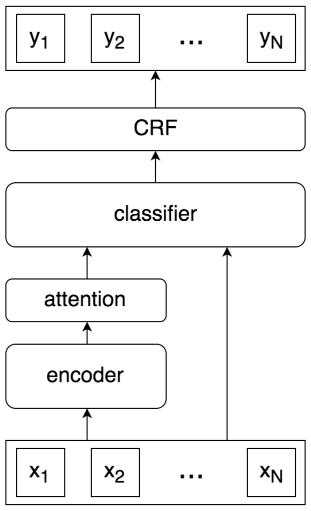
Author	Components	Additional Features	SemEval 2014 Performance
(CHERNYSHEVICH, 2014) #	CRF	NER + POS + Sentiment Lexicon	L: 74.55%, R: 79.62%
(TOH; WANG, 2014) #	CRF	POS + Name list + Sentiment Lexicon + Word dependencies	L: 73.78%, R: 84.01%
(GIANNAKOPOULOS *et al.*, 2017)	BLSTM + CRF	-	L: 77.96%, R: 84.12%
(WANG *et al.*, 2016)	RecNN + CRF	POS + Name list + Sentiment Lexicon	L: 78.42%, R: 84.93%
(LI *et al.*, 2018)	LTSM + Truncated History-Attention (THA) and Selective Transformation Network (STN)	-	L: 79.52%, R: 85.61%
(WANG *et al.*, 2017)	GRU + Coupled Multi-layer Attentions (CMLA)	-	L: 80.17%, R: 85.79%
(PORIA *et al.*, 2016a)	CNN + CRF	Domain Specific Embeddings + POS (manually selected) + Linguistic Patterns + Sentiment Lexicon	L: 82.32%, R: 87.17%

4 ASPECT TERM EXTRACTION VIA BLSTM-CRF WITH ATTENTION ON WORD DEPENDENCIES

4.1 Overview

In this chapter, we present our proposed end-to-end neural network architecture: a BLSTM encoder with an attention mechanism to extract information from the word dependencies in the sentence, and a classifier composed also by a BLSTM but with a CRF layer on top, that uses the encoder information to extract the aspect terms from the sentences. Figure 15 shows an overview of the architecture proposed.

Figure 15 – Overview of the proposed architecture.



Our architecture was based on the encoder-decoder architecture proposed in (CHO *et al.*, 2014b) and (SUTSKEVER *et al.*, 2014). This architecture has two main modules: an encoder, responsible for learning a vector representation for a sequence of tokens, and a decoder, responsible for generating a sequence of tokens from a vector that represents an encoded sequence. In general, this architecture is used to build models that can learn how to transform a sequence of tokens into another one. Our model is based on the encoder-decoder architecture, however, its modules do not necessarily have the same roles. In this work, we use an attention mechanism on the encoder as a strategy to add into the model the information given by the word dependencies between the words in a sentence, and instead of a decoder, we have a classifier that outputs the most likely label sequence for the sentence.

Our BLSTM-CRF classifier was built based on the architecture proposed by (LAMPLE *et al.*, 2016), due to its state-of-the-art performance when applied to sequence labelling problems.

The next sections explain with detail the encoder and the classifier structures.

4.2 Encoder

The encoder is composed by a BLSTM, that takes as input the word embeddings and the POS tag vectors of each word in the sentence and computes a sentence representation.

Given a sentence s of size n, the word embeddings of each word $(x_1, x_2, \cdots, x_n)$ and the POS tag vectors of each word $(p_1, p_2, \cdots, p_n)$, the BLSTM (encoder) takes as input the word embedding concatenated with the POS tag vector of each word $(x_1;p_1, x_2;p_2, \cdots, x_n;p_n)$ and computes the hidden states outputs $(h_1, h_2, \cdots, h_n)$ where:

$$h_j = [\overrightarrow{h_j}; \overleftarrow{h_j}] \tag{4.1}$$

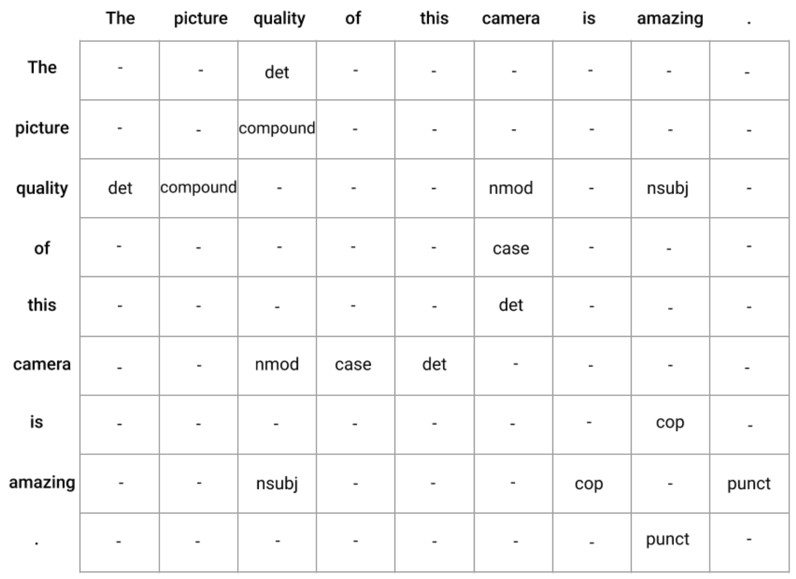
that is, h_j is the concatenation of the forward hidden state $\overrightarrow{h_j}$ and the backward hidden state $\overleftarrow{h_j}$ computed by the forward and backward LSTMs respectively.

4.3 Attention Mechanism and Word Dependencies

The attention mechanism is added to the encoder to compute different sentence representations, or context vectors, for each word in the sentence based on the word dependencies information. Instead of using the same context vector, or sentence representation, generated by the encoder for all words, the attention mechanism computes a different context vector for each word based on the grammatical relation between the words in the sentence, thus, when performing the classification, every word will have their specific context vector.

The word dependencies can be represented as a matrix, where each entry in the matrix represents the word dependency between two words in the sentence. Figure 16 illustrates the word dependencies matrix for the sentence *"The picture quality of this camera is amazing."*. The attention weights are computed using an alignment model that takes as input the word dependencies vectors of each word. In this work we use a Multilayer Perceptron (MLP) with two hidden layers to compute the attention weights, using as input the dependencies vectors.

Figure 16 – Word dependencies matrix for the sentence *“The picture quality of this camera is amazing.”*



	The	picture	quality	of	this	camera	is	amazing	.
The	-	-	det	-	-	-	-	-	-
picture	-	-	compound	-	-	-	-	-	-
quality	det	compound	-	-	-	nmod	-	nsubj	-
of	-	-	-	-	-	case	-	-	-
this	-	-	-	-	-	det	-	-	-
camera	-	-	nmod	case	det	-	-	-	-
is	-	-	-	-	-	-	-	cop	-
amazing	-	-	nsubj	-	-	-	cop	-	punct
.	-	-	-	-	-	-	-	punct	-

Given a sentence s of size n, the words $(w_1, w_2, \cdots, w_n)$ of the sentence, the word dependency vectors $(d_{i,1}, d_{i,2}, \cdots, d_{i,n})$ of the i^{th} word in the sentence, the scores $e_{i,j}$ of how well the words w_i and w_j match are given by:

$$e_{i,j} = W_a(\tanh(W_{a2}\tanh(W_{a1}d_{i,j} + b_{a1}) + b_{a2})) + b_a \tag{4.2}$$

where (W_a, W_{a1}, W_{a2}) are the weight matrices and (b_a, b_{a1}, b_{a2}) the bias vectors of a two hidden layer MLP.

The attention weights are then computed using a softmax:

$$\alpha_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k=1}^{n}\exp(e_{i,k})} \tag{4.3}$$

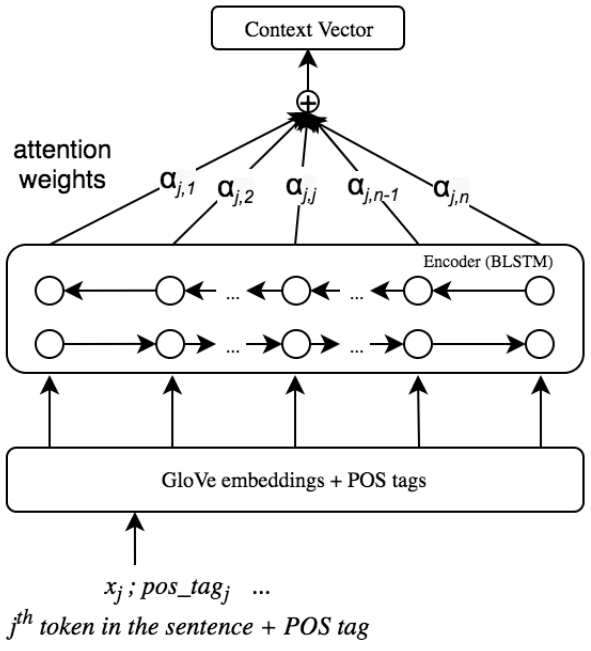
And, finally, the context vector c_i of the ith word in the sentence is given by:

$$c_i = \sum_{j=1}^{n} \alpha_{i,j} h_j \tag{4.4}$$

where h_j are the hidden states outputs of the BLSTM encoder.

The encoder structure with the attention mechanism is represented in the Figure 17.

Figure 17 – Representation of the encoder with the attention mechanism.



4.4 BLSTM-CRF classifier

The classifier is composed by a BLSTM, a fully connected layer and a CRF layer. The BLSTM takes as input the word embeddings, the POS tag vectors and the context vectors generated by the encoder with attention, and output representations using those information.

Given a sentence s of size n, the word embeddings of each word $(x_1, x_2, \cdots, x_n)$, the POS tag vectors of each word $(p_1, p_2, \cdots, p_n)$, and the context vectors $(c_1, c_2, \cdots, c_n)$ computed by the encoder with attention, the BLSTM (encoder) takes as input the word embedding concatenated with the POS tag vector and context vector of each word $(x_1; p_1; c_1, x_2; p_2; c_2, \cdots, x_n; p_n; c_n)$ and computes the hidden states outputs $(h_1, h_2, \cdots, h_n)$ where:

$$h_j = [\overrightarrow{h_j}; \overleftarrow{h_j}] \tag{4.5}$$

The BLSTM output representations are then mapped to the three possible labels: “I”, “O” or “B”, through a fully connected layer.

Finally, the CRF layer takes as input the output vectors of the fully connected layer and computes the most likely label sequence for the sentence, exploring the latent correlations between the labels.

Let y' be a possible label sequence, and X the input representations of the words in the sentence. The CRF layer first computes the label predictions score s using the fully connected layer output matrix H and the learnable transition scores matrix T:

$$s(X,y') = \sum_{i=1}^{n} T_{y'_i,y'_{i+1}} + \sum_{i=1}^{n} H_{x_i,y'_i} \tag{4.6}$$

A probability for a label sequence y' is calculated using a softmax over all the possible label sequences:

$$p(y'|X) = \frac{e^{s(X,y')}}{\sum_{\tilde{y}\in Y_X} e^{s(X,\tilde{y})}} \tag{4.7}$$

where Y_X is the set of all possible label sequences.

In the training, the CRF layer maximizes the log-probability of the correct label sequence:

$$\log(p(y'|X)) = s(X,y') - \log(\sum_{\tilde{y}\in Y_X} e^{s(X,\tilde{y})}) \tag{4.8}$$

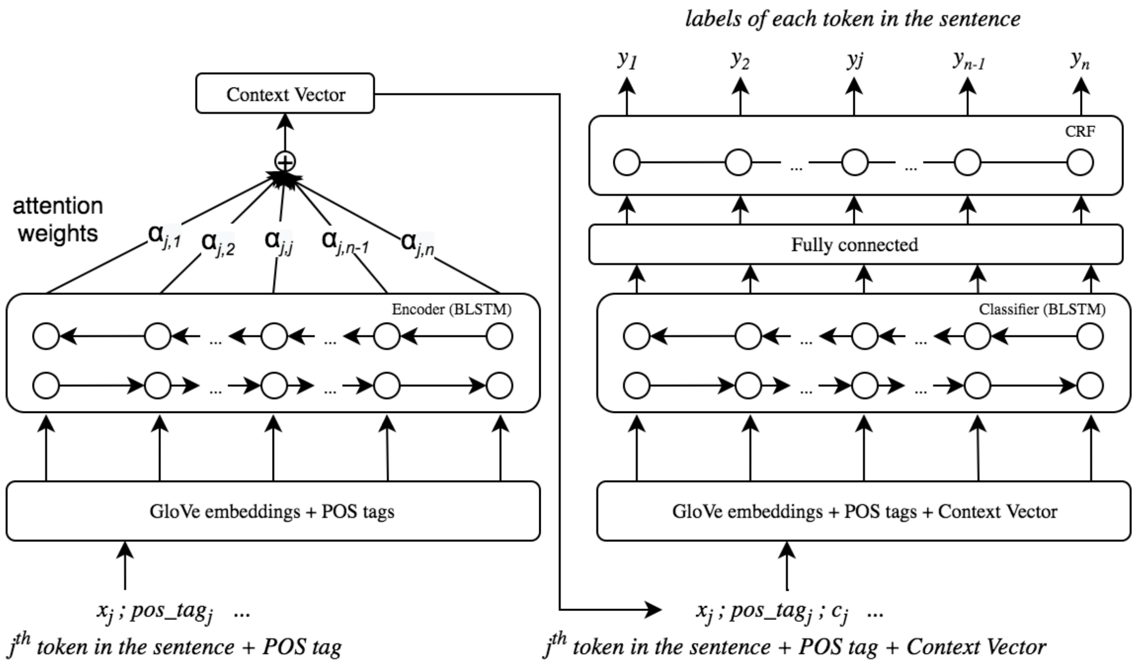
In the decoding phase, it returns the output label sequence y^* with the maximum score:

$$y^* = \underset{\tilde{y}\in Y_X}{\operatorname{argmax}}\, s(X,\tilde{y}) \tag{4.9}$$

4.5 Final architecture

Figure 18 shows a detailed scheme of the architecture proposed, composed by the encoder with the attention mechanism and the classifier structures.

Figure 18 – The neural network architecture proposed. The attention mechanism uses the word dependencies information to weight the encoder hidden states, and combine them to generate a different context vector for each word in the sentence.



labels of each token in the sentence
y_1
y_2
yj
y_{n-1}
y_n
Context Vector
CRF
attention weights
$\alpha_{j,1}$
$\alpha_{j,2}$
$\alpha_{j,j}$
$\alpha_{j,n-1}$
$\alpha_{j,n}$
Fully connected
Encoder (BLSTM)
Classifier (BLSTM)
GloVe embeddings + POS tags
GloVe embeddings + POS tags + Context Vector
x_j ; pos_tag_j ...
j^{th} token in the sentence + POS tag
x_j ; pos_tag_j ; c_j ...
j^{th} token in the sentence + POS tag + Context Vector

5 EXPERIMENTS AND RESULTS

5.1 Datasets

The datasets used in this work to assess the models were: the Laptop and Restaurant domain datasets from the SemEval 2014 competition[1] and the Restaurant domain dataset, Subtask 2, from the SemEval 2016 competition[2]. The datasets contain reviews about restaurants and laptops written by users, with manual annotations indicating the aspect terms present in each sentence with their sentiment associated. In this work, the datasets were tagged using IOB2 tags and their statistics are described in Table 2.

Table 2 – Number of training sentences, test sentences, and aspect terms present in the SemEval 2014 and 2016 datasets.

Domain	Training	Test	Total	# Aspect terms
Laptop (2014)	3,041	800	3,841	2,373
Restaurant (2014)	3,045	800	3,845	3,699
Restaurant (2016)	2,000	676	2,676	2,530

The datasets were pre-processed and annotated with POS tags generated using the Stanford POS tagger (TOUTANOVA *et al.*, 2003) and word dependencies generated by the Stanford Dependencies (MARNEFFE; MANNING, 2008). In total, there were 41 different types of POS tags and 39 different types of word dependencies in the datasets. It is important to emphasize that we did not manually select a subset of those features, we used all the existent POS tags and word dependencies and let the model select those that are most relevant during the training phase.

As word representations, we used the 300d GloVe embeddings (PENNINGTON *et al.*, 2014) trained on 6B tokens. Word embeddings are distributed representations of text, which encode semantic and syntactic properties of words.

5.2 Experiment setup

To define the number of LSTM cells units used on the encoder and classifier, the ideal number of epochs to train the ATE problem, and the dropout rate, we randomly sampled 10% of

[1] http://alt.qcri.org/semeval2014/task4/index.php?id=data-and-tools
[2] http://alt.qcri.org/semeval2016/task5/index.php?id=data-and-tools

the datasets and took the average performance of 3 runs for validation. The hyperparameters used on the experiments are described in Table 3.

Table 3 – The hyperparameters used for each model and dataset.

Domain	Model	# Epochs	# Cells (Enc.)	# Cells (Class.)
Laptop (2014)	Enc-BLSTM-CRF	30	128	256
	Enc-BLSTM-CRF+POS	32	128	256
	AttWD-BLSTM-CRF+POS	56	128	256
Restaurant (2014)	Enc-BLSTM-CRF	22	128	256
	Enc-BLSTM-CRF+POS	16	128	256
	AttWD-BLSTM-CRF+POS	46	128	256
Restaurant (2016)	Enc-BLSTM-CRF	58	256	256
	Enc-BLSTM-CRF+POS	37	256	256
	AttWD-BLSTM-CRF+POS	31	256	256

The models were trained using the Adam algorithm(KINGMA; BA, 2014) with a learning rate of 0.01 and dropout rate of 20%(HINTON *et al.*, 2012) on LSTM layers.

5.3 Evaluation Metrics

The evaluation metrics used to assess the ATE task are the precision, recall and F1 score, that is the harmonic mean between the precision and recall. For multi word (compound) aspect terms, if the model assigns a wrong label for any word in the aspect, then it is accounted as a misclassification. The precision, recall and F1 score are defined as:

$$precision = \frac{true\_positive}{true\_positive + false\_positive} \tag{5.1}$$

$$recall = \frac{true\_positive}{true\_positive + false\_negative} \tag{5.2}$$

$$F1 = 2 \times \left(\frac{precision \times recall}{precision + recall} \right) \tag{5.3}$$

5.4 Experiments

We conduct a few experiments to assess some aspects of our proposed architecture and validate its relevance for the ATE problem: we analyze the importance of the additional features for the ATE problem and for sentences containing out of vocabulary words; we compare the results obtained using our proposed architecture with state-of-the-art methods; and we plotted the attention weights computed by our model to understand how the model makes use of the word dependencies information to perform the ATE task.

5.4.1 Additional features importance

We assessed three different models based on our proposed architecture to study the importance of the POS tag feature and the attention mechanism using word dependencies for the problem of aspect term extraction:

- **Enc-BLSTM-CRF**: the encoder and the BLSTM-CRF classifier using no additional features.
- **Enc-BLSTM-CRF+POS**: the encoder and the BLSTM-CRF classifier with the POS tag feature.
- **AttWD-BLSTM-CRF+POS**: the encoder with the attention mechanism on word dependencies, and the BLSTM-CRF classifier, along with the POS tag feature.

Table 4 shows the evaluation metrics obtained for each model on the test sets. The results showed are the average performance of 10 runs. Figures 19, 20 and 21 are box plots of the F1 scores obtained after 10 executions for each model, using the Laptop (2014), Restaurant (2014) and Restaurant (2016) domain datasets respectively.

Table 4 – Results obtained using the three different models. P stands for precision, R for recall and F1 for F1 score.

Domain	Model	P	R	F1
Laptop (2014)	Enc-BLSTM-CRF	89.31%	73.59%	80.69%
	Enc-BLSTM-CRF+POS	88.50%	**74.90%**	81.12%
	AttWD-BLSTM-CRF+POS	**89.51%**	74.78%	**81.47%**
Restaurant (2014)	Enc-BLSTM-CRF	89.52%	85.50%	87.46%
	Enc-BLSTM-CRF+POS	88.97%	**86.61%**	87.75%
	AttWD-BLSTM-CRF+POS	**89.81%**	86.00%	**87.86%**
Restaurant (2016)	Enc-BLSTM-CRF	72.35%	70.07%	71.15%
	Enc-BLSTM-CRF+POS	72.15%	72.23%	72.13%
	AttWD-BLSTM-CRF+POS	**73.05%**	**73.15%**	**73.04%**

Figure 19 – F1 score statistics for the Laptop (2014) domain dataset.

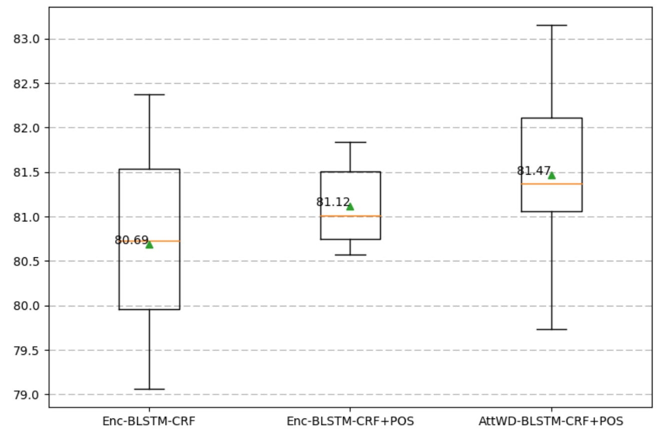


Figure 20 – F1 score statistics for the Restaurant (2014) domain dataset.

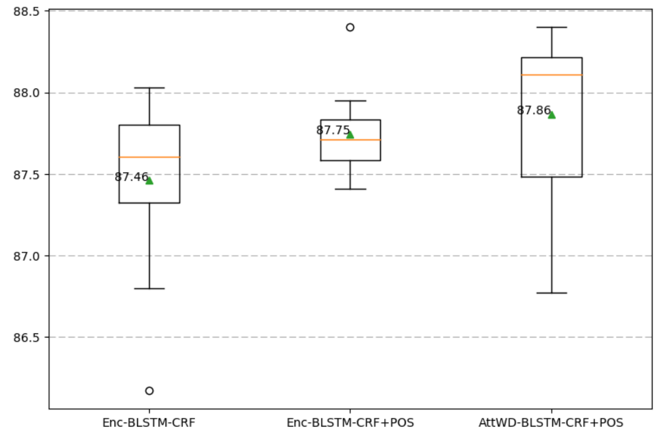
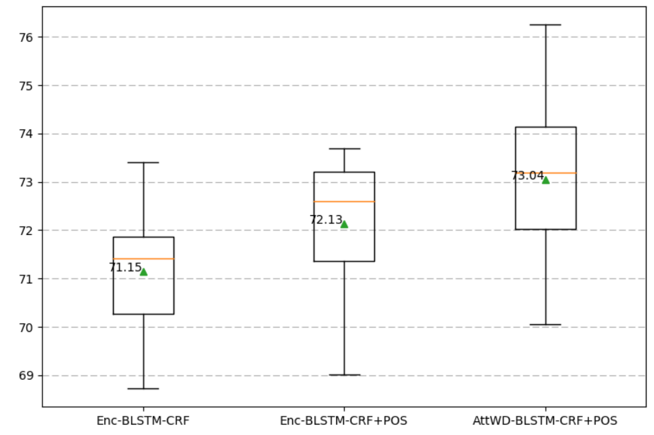


Figure 21 – F1 score statistics for the Restaurant (2016) domain dataset.



From the results showed in Table 4, we claim that both the POS tag feature and the attention mechanism with word dependencies are important to improve the recall metric, increasing the model capability of identifying aspect terms in the sentences, reducing the number of false negatives.

Figures 19, 20 and 21 shows that, despite having less consistent results when compared with the model using using only the POS tag feature, the model using word dependencies was able to achieve higher F1 scores in all the datasets.

5.4.2 Comparison with state-of-the-art

We compare our proposed model (AttWD-BLSTM-CRF+POS) with state-of-the-art models for the ATE task:

– **BLSTM-CRF**: a BLSTM-CRF classifier from (GIANNAKOPOULOS *et al.*, 2017).
– **Poria**: a deep convolutional neural network combined with language rules, that uses filtered POS tags and a lexicon as additional features, from (PORIA *et al.*, 2016a).
– **Li**: a framework for ATE that uses truncated history attention and a selective transformation network to incorporate opinion information, from (LI *et al.*, 2018).
– **IHS_RD**: a model that uses the IHS Goldfire linguistic processor and a CRF, the winner of the SEMEVAL 2014 challenge, ATE subtask on the Laptop domain (CHERNYSHEVICH, 2014).
– **DLIREC**: a CRF classifier with manually engineered features, the winner of the SEMEVAL 2014 challenge, ATE subtask on the Restaurant domain (TOH; WANG, 2014).
– **NLANGP**: a RNN-CRF classifier with manually engineered features, the winner of the SEMEVAL 2016 challenge, ATE subtask on the Restaurant domain (TOH; SU, 2016).

Table 5 shows the F1 scores obtained using our proposed architecture and the state-of-the-art methods. The results for our competitors were copied from their papers.

Table 5 – Comparison between the F1 scores obtained using our architecture and state-of-the-art methods. '-' indicates the results were not available in their papers.

Model	Laptop (2014)	Restaurant (2014)	Restaurant (2016)
IHS_RD	74.55%	79.62%	-
DLIREC	73.78%	84.01%	-
NLANGP	-	-	72.34%
BLSTM-CRF	77.96%	84.12%	-
Li	79.52%	85.61%	**73.61%**
Poria	**82.32%**	87.17%	-
AttWD-BLSTM-CRF+POS (our)	81.47%	**87.86%**	73.04%

Table 6 – Comparison between different approaches and features used to perform ATE with our architecture, and the performance obtained by them on the SEMEVAL 2014 datasets. Classical approaches that do not use DL are marked with (#). L stands for the Laptop dataset and R for the Restaurant dataset.

Model	Components	Additional Features	SemEval 2014 Performance
IHS_RD #	CRF	NER + POS + Sentiment Lexicon	L: 74.55%, R: 79.62%
DLIREC #	CRF	POS + Name list + Sentiment Lexicon + Word dependencies	L: 73.78%, R: 84.01%
BLSTM-CRF	BLSTM + CRF	-	L: 77.96%, R: 84.12%
(WANG *et al.*, 2016)	RecNN + CRF	POS + Name list + Sentiment Lexicon	L: 78.42%, R: 84.93%
LI	LTSM + Truncated History-Attention (THA) and Selective Transformation Network (STN)	-	L: 79.52%, R: 85.61%
(WANG *et al.*, 2017)	GRU + Coupled Multi-layer Attentions (CMLA)	-	L: 80.17%, R: 85.79%
Poria	CNN + CRF	Domain Specific Embeddings + POS (manually selected) + Linguistic Patterns + Sentiment Lexicon	L: **82.32%**, R: 87.17%
AttWD-BLSTM-CRF+POS (our)	BLSTM + Attention Mechanism + CRF	POS + Word Dependencies	L:81.47%, R:**87.86%**

Table 6 compares the model components, additional features and the performance obtained on the SEMEVAL 2014 datasets between the state-of-the-art methods and our proposed architecture. Our model achieved competitive results when compared with other state-of-the-art models, but using only the POS tag and word dependencies features, and without manually selecting a subset of them.

5.4.3 Out of vocabulary (OOV) words

We also evaluated the importance of the additional POS tag and word dependencies features on handling sentences that contain at least one out of vocabulary (OOV) word. Table 7 compares the results obtained using our architecture with and without the additional features for sentences in the test sets that have at least one OOV word.

Table 7 – Comparison of the models with and without additional features on sentences with OOV words. P stands for precision, R for recall and F1 for F1 score.

Domain	# Sentences	Model	P	R	F1
Laptop (2014)	25	Enc-BLSTM-CRF	81.82%	46.15%	59.02%
		AttWD-BLSTM-CRF+POS	**91.67%**	**56.41%**	**69.84%**
Restaurant (2014)	103	Enc-BLSTM-CRF	81.68%	**77.83%**	79.71%
		AttWD-BLSTM-CRF+POS	**86.17%**	76.42%	**81.00%**
Restaurant (2016)	84	Enc-BLSTM-CRF	63.74%	52.25%	57.43%
		AttWD-BLSTM-CRF+POS	**65.35%**	**59.46%**	**62.26%**

From Table 7, we can attest that the additional POS tag and word dependencies features improves the results on sentences with OOV words.

5.4.4 Attention visualization

We also analyzed the attention weights computed by the AttWD-BLSTM-CRF+POS model.

Figures 22 and 23 shows, respectively, the word dependencies and attention weights computed for the sentence *"The portions of the food that came out were mediocre."*, where the aspect term is *"The portions of the food"*. Figures 24 and 25 shows the dependencies and weights computed for another sentence: *"It is super fast and has outstanding graphics."*, where the aspect term is *"graphics"*.

Figure 22 – Word dependencies generated for the sentence *"The portions of the food that came out were mediocre."*.

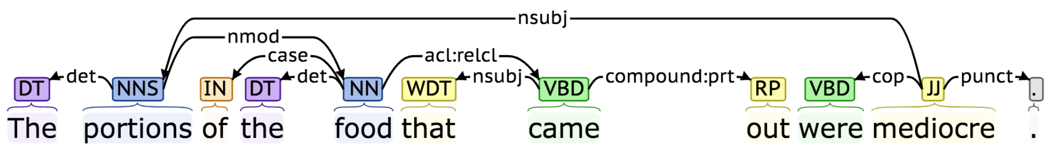


Figure 23 – Attention weights computed for the sentence *“The portions of the food that came out were mediocre.”*.

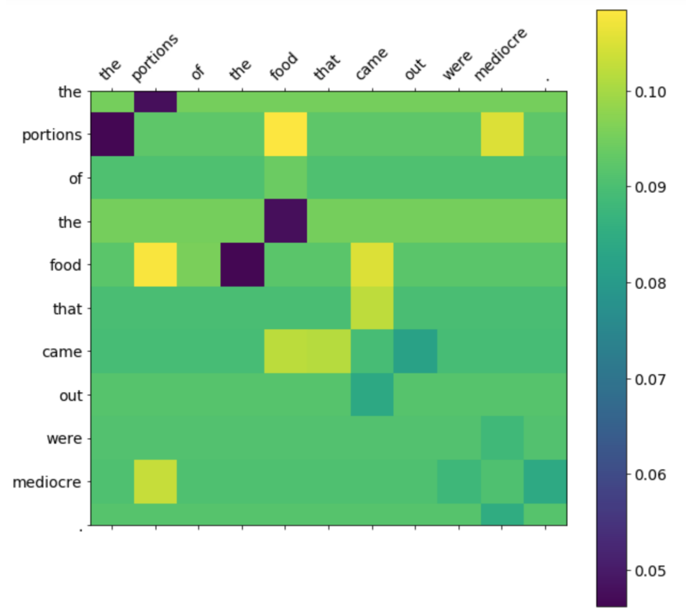


Figure 24 – Word dependencies generated for the sentence *“It is super fast and has outstanding graphics.”*.

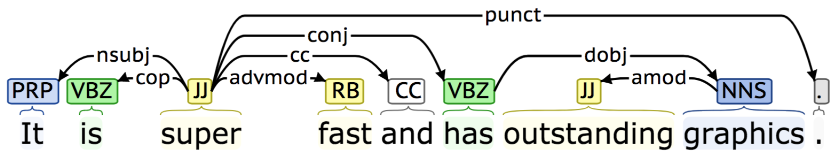
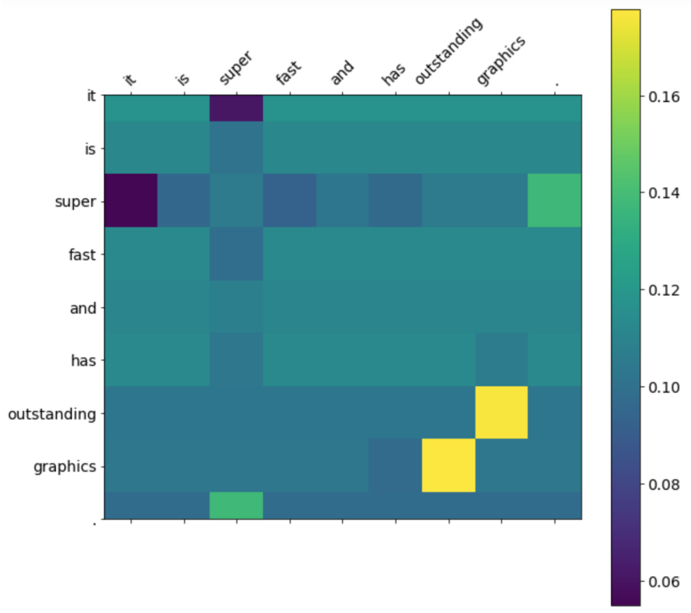


Figure 25 – Attention weights computed for the sentence *“It is super fast and has outstanding graphics.”*.



For the first sentence *"The portions of the food that came out were mediocre."*, we can see in figure 23 that the attention mechanism gave more importance to the dependencies *nsubj* (nominal subject), between the words *"portions"* and *"mediocre"*, and *nmod* (nominal modifier), between the words *"portions"* and *"food"*, computing higher weights for them. It implies that the model considers the relations *nominal subject* and *nominal modifier* relevant to identify aspect words in the sentence. The same behaviour can be seen on the second sentence *"It is super fast and has outstanding graphics."* in figure 25, where higher weights were computed for the relation *amod* (adjectival modifier) between the words *"graphics"* and *"outstanding"*, supporting our hypothesis that the word dependency feature is important to the problem of aspect extraction.

6 CONCLUSION

In this work we presented a deep neural network architecture to perform the Aspect Term Extraction task, using only two additional features: part-of-speech (POS) tags and the grammatical dependencies between words in a sentence.

Analyzing the state-of-the-art works for the ATE problem, we concluded that deep neural networks models have the best performance, however, in a scarce labeled data scenario, these architectures are not able by themselves to perform the ATE task accurately, and additional features are important to improve these models.

Thus, we incorporated the POS tags and the word dependencies as additional features in our model, but unlike other works (PORIA *et al.*, 2016a; WANG *et al.*, 2016), we did not manually filter a subset of these additional features, we let the model select those that are most relevant in the training phase. We also not used linguistic rules or lexicons, i.e. we aimed to perform minimal feature engineering.

The novelty in our architecture is the use of an encoder, a common structure used in Neural Machine Translation (CHO *et al.*, 2014a; SUTSKEVER *et al.*, 2014), with an attention mechanism that allows the use of the grammatical relation between words as an additional feature. This feature is important to the problem of aspect term extraction because aspect terms are usually nominal subjects (syntactic subjects) with an adjective associated, and it also helps to identify multiple word aspect terms, using the compound dependency between nouns in a noun phrase.

The experiments results showed that the additional features used are relevant to the ATE problem, and can help the model to identify aspect terms in sentences with out of vocabulary (OOV) words. Also, our architecture obtained competitive results when compared to other state-of-the-art methods.

As future works, we plan to evaluate segment representations different from the IOB2 tagging format used, like the BILOU (KONKOL; KONOPÍK, 2015), and incorporate character embeddings as an additional input feature. We also aim to test different alignment score functions for the attention mechanism, and replace the LSTMs used in the encoder and classifier by more efficient structures, like Gated Recurrent Units (GRU).

REFERENCES

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. **arXiv preprint arXiv:1409.0473**, 2014.

BENGIO, Y.; SIMARD, P.; FRASCONI, P. *et al.* Learning long-term dependencies with gradient descent is difficult. **IEEE transactions on neural networks**, v. 5, n. 2, p. 157–166, 1994.

CHERNYSHEVICH, M. Ihs r&d belarus: Cross-domain extraction of product features using crf. In: **Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)**. [*S. l.*: *s. n.*], 2014. p. 309–313.

CHO, K.; MERRIËNBOER, B. V.; GULCEHRE, C.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. **arXiv preprint arXiv:1406.1078**, 2014.

CHO, K.; MERRIËNBOER, B. V.; GULCEHRE, C.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. **arXiv preprint arXiv:1406.1078**, 2014.

Christopher Olah. **Understanding LSTM Networks**. 2015. Disponível em: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

DO, H. H.; PRASAD, P.; MAAG, A.; ALSADOON, A. Deep learning for aspect-based sentiment analysis: a comparative review. **Expert Systems with Applications**, Elsevier, v. 118, p. 272–299, 2019.

ELMAN, J. L. Finding structure in time. **Cognitive science**, Wiley Online Library, v. 14, n. 2, p. 179–211, 1990.

GIANNAKOPOULOS, A.; MUSAT, C.; HOSSMANN, A.; BAERISWYL, M. Unsupervised aspect term extraction with b-lstm & crf using automatically labelled datasets. **arXiv preprint arXiv:1709.05094**, 2017.

GRAVES, A.; SCHMIDHUBER, J. Framewise phoneme classification with bidirectional lstm and other neural network architectures. **Neural networks**, Elsevier, v. 18, n. 5-6, p. 602–610, 2005.

HASPELMATH, M. Word classes and parts of speech. 2001.

HINTON, G. E.; SRIVASTAVA, N.; KRIZHEVSKY, A.; SUTSKEVER, I.; SALAKHUTDINOV, R. R. Improving neural networks by preventing co-adaptation of feature detectors. **arXiv preprint arXiv:1207.0580**, 2012.

HOCHREITER, S. Untersuchungen zu dynamischen neuronalen netzen. **Diploma, Technische Universität München**, v. 91, n. 1, 1991.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HU, M.; LIU, B. Mining and summarizing customer reviews. In: **Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining**. [*S. l.*: *s. n.*], 2004. p. 168–177.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. **arXiv preprint arXiv:1412.6980**, 2014.

KONKOL, M.; KONOPÍK, M. Segment representations in named entity recognition. In: SPRINGER. **International Conference on Text, Speech, and Dialogue**. [*S. l.*], 2015. p. 61–70.

LAKKARAJU, H.; SOCHER, R.; MANNING, C. Aspect specific sentiment analysis using hierarchical deep learning. In: **NIPS Workshop on Deep Learning and Representation Learning**. [*S. l.*: *s. n.*], 2014.

LAMPLE, G.; BALLESTEROS, M.; SUBRAMANIAN, S.; KAWAKAMI, K.; DYER, C. Neural architectures for named entity recognition. **arXiv preprint arXiv:1603.01360**, 2016.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015.

LI, X.; BING, L.; LI, P.; LAM, W.; YANG, Z. Aspect term extraction with history attention and selective transformation. **arXiv preprint arXiv:1805.00760**, 2018.

LIU, B.; ZHANG, L. A survey of opinion mining and sentiment analysis. In: **Mining text data**. [*S. l.*]: Springer, 2012. p. 415–463.

MARCUS, M.; SANTORINI, B.; MARCINKIEWICZ, M. A. Building a large annotated corpus of english: The penn treebank. 1993.

MARNEFFE, M.-C. D.; MANNING, C. D. **Stanford typed dependencies manual**. [*S. l.*], 2008.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: **Empirical Methods in Natural Language Processing (EMNLP)**. [*S. n.*], 2014. p. 1532–1543. Disponível em: http://www.aclweb.org/anthology/D14-1162.

PIRYANI, R.; GUPTA, V.; SINGH, V. K. Movie prism: A novel system for aspect level sentiment profiling of movies. **Journal of Intelligent & Fuzzy Systems**, IOS Press, v. 32, n. 5, p. 3297–3311, 2017.

PORIA, S.; CAMBRIA, E.; GELBUKH, A. Aspect extraction for opinion mining with a deep convolutional neural network. **Knowledge-Based Systems**, Elsevier, v. 108, p. 42–49, 2016.

PORIA, S.; CHATURVEDI, I.; CAMBRIA, E.; BISIO, F. Sentic lda: Improving on lda with semantic similarity for aspect-based sentiment analysis. In: IEEE. **2016 international joint conference on neural networks (IJCNN)**. [*S. l.*], 2016. p. 4465–4473.

RANA, T. A.; CHEAH, Y.-N. Aspect extraction in sentiment analysis: comparative analysis and survey. **Artificial Intelligence Review**, Springer, v. 46, n. 4, p. 459–483, 2016.

RAVI, S.; LAROCHELLE, H. Optimization as a model for few-shot learning. 2016.

SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. In: **Advances in neural information processing systems**. [*S. l.*: *s. n.*], 2014. p. 3104–3112.

TOH, Z.; SU, J. Nlangp at semeval-2016 task 5: Improving aspect based sentiment analysis using neural network features. In: **Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)**. [*S. l.*: *s. n.*], 2016. p. 282–288.

TOH, Z.; WANG, W. Dlirec: Aspect term extraction and term polarity classification system. In: **Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)**. [*S. l.*: *s. n.*], 2014. p. 235–240.

TOUTANOVA, K.; KLEIN, D.; MANNING, C. D.; SINGER, Y. Feature-rich part-of-speech tagging with a cyclic dependency network. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1**. [*S. l.*], 2003. p. 173–180.

WANG, W.; PAN, S. J.; DAHLMEIER, D.; XIAO, X. Recursive neural conditional random fields for aspect-based sentiment analysis. **arXiv preprint arXiv:1603.06679**, 2016.

WANG, W.; PAN, S. J.; DAHLMEIER, D.; XIAO, X. Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In: **Thirty-First AAAI Conference on Artificial Intelligence**. [*S. l.*: *s. n.*], 2017.