Balancing between Holistic and Cumulative Sentiment Classification

Pantelis Agathangelou*, Ioannis Katakis

Department of Computer Science, School of Sciences and Engineering, University of Nicosia, CY-2417, Nicosia, Cyprus

Abstract

Sentiment analysis is a fast-accelerating discipline that develops algorithms for knowledge discovery from opinionated content. The challenges however, when it comes to analyzing user reviews are plenty. Bad-quality, informal use of language and lack of labels, are only a few obstacles. Most importantly, users, consciously or subconsciously, use different approaches for expressing their opinion about a product or a service. Some of them go sentence by sentence mentioning some positive and negative aspects whereas others provide a mixed piece of text where the reader is supposed to see the big picture to understand the message. In this work, we propose a novel neural network that deals with both situations. Our method, by combining convolutional, recurrent and attention neural networks can extract rich linguistic patterns that reveal the user's sentiment towards the entity under review. We evaluate our method in nine datasets that represent both binary and multi-class classification tasks. Experimental evaluation indicates that our method outperforms well-established deep learning approaches. Our approach outperformed the competitive methods in 8 out of 9 cases.

Keywords: Sentiment Analysis, Opinion Mining, Text Classification

*Corresponding author

Email addresses: agathangelou.p@live.unic.ac.cy (Pantelis Agathangelou), katakis.i@unic.ac.cy (Ioannis Katakis)

1. Introduction

Online content like user opinionated reviews, contains information that if exploited appropriately may provide commercial and research value. Due to the volume of data, the content cannot be parsed or analyzed manually. Hence, the development of efficient tools to analyze automatically such information sources becomes a necessity. Sentiment Analysis of language is one of the most discussed topics in Text Analysis.

The challenges however, when it comes to modeling such language are numerous. Opinionated content consists of linguistic terms which may extend to a few thousands. Moreover, these terms may appear anywhere in a sentence and despite the syntactic and semantic constraints, the possible combinations of words in a vocabulary of size $|V|$ in sentences of size s are at the range of $|V|^s$. Another issue that directly links to problems like sentiment analysis is disambiguity. In this case, sentences that have very similar words and word ordering might express different sentiment in different contexts.

In modeling language, many approaches have been developed. First efforts included n-gram statistical [1] or neural probabilistic models [2]. We refer to n-grams as shallow or template-learning models. In general, neural models produce better generalization mainly because of the utilization of distributed word vectors. More recent work mainly focuses on neural networks that target the sequential nature of words and sentences. In this line of work three types of models can be identified: i) convolutional [3, 4, 5, 6], ii) recurrent [7, 8, 9, 10] and iii) recursive neural networks [11, 12]. Many of these approaches have demonstrated superior accuracy in sentiment analysis settings. One point however where these models lack, is the ability to correctly identify mixed expressed sentiment. Consider for example the following review at Figure 1 that comes from a real dataset[1] and it includes a review of a catering service.

In parentheses we note the sentiment of the expressed sentence or phrase ('+' for positive and '−' for negative). Color (green/red/blue) also represents sentiment (positive/negative/neutral). Assigning sentiment to sentences (or phrases) is a trivial task in most cases. Trying to assess the sentiment for a large piece of text (like a product review) however, might be a little bit more challenging. This is mainly because language might include a mixed distribution of sentiment.

[1] See Section 4, YELP dataset

> “The location is excellent (+). The food is mediocre, and milder than they advertise (−). The wait staff is polite and bland (+). At several opportunities, they were missing for more than 5 mins (±). The bill will be higher than anyone expects (−).”

Figure 1: An Opinion Example From a Real Dataset and the Process of Polarity Disambiguation.

Cumulative Classification vs Holistic. In the example above, we can see that there are two possible ways of assessing sentiment in the text. One is to assign the final classification considering all partial classifications (how many positive sentences vs how many negative), whereas the other is to assign a classification to the sentiment of the whole snippet (how positive or negative a review is assessed by a user). We refer to the first one as ‘cumulative’[2] and to the latter as ‘holistic’ [13]. Model architectures developed so far exploit only the first or the second type in order to draw conclusions about the sentiment expressed in text. This is however problematic since, as we will see, both approaches can contribute towards a more efficient sentiment classification. In this paper, in order to alleviate the above limitation, we introduce an attention based convolution network to the widely used recurrent deep learning model. A key component of our solution is a classical layer that we integrate between the outputs of a modified attention layer that exploits a bi-directional recurrent network and the final prediction layer. Since recurrent networks elaborate semantic content sequentially, subsequent outputs from such networks provide both local and global information. We exploit this attribute by forcing the optimization algorithm to identify the balance between ‘holistic’ and ‘cumulative’ users in the classification task via a tunable hyper-parameter that we call ‘Balancing Factor’. By considering holistic and cumulative content our method improves in accuracy. The contributions of this paper can be summarized in the following points.

[2] In the literature the term is also met as ‘analytic’

- We present a novel and effective Deep Neural Network for modeling opinionated content.
- We introduce a Neural Hyper-Parameter that joins mixed content motifs (Holistic and Cumulative).
- We introduce an Attention Layer that combines corpus optimization along with document optimization.
- We experiment over a set of nine benchmark datasets and demonstrate that our proposed approach outperformed the competitive methods in most cases.

The rest of the paper is structured as follows. At Section 2, we present the related work. Next, the proposed method is described in detail at Section 3, while Section 4 presents the hyper-parameters and training details part of the paper. Section 5 presents the evaluation results of two classification problems, Sentiment (Subsection 5.1) and Question type classification (Subsection 5.2). The contribution of the k-max pooling operation is presented at Section 6. At Section 7 our method is tested on a multi-domain sentiment classification. At Section 8 we include the error analysis where we identify the predictive characteristics of our method. Finally, we present the results and our discussion in Section 9.

This work builds on top of our previous work presented in [14]. At that work an innovative neural method tackled the document level sentiment analysis task and compared its performance against other well established machine learning methods. The introduction of the global sentence embedding and the output window size hyper-parameter were a few of the main contributions. This work extends and improves the work presented in [14], in the following ways:

1. We improve the bottom part of our model, the convolution layer. In the new setting the convolution layer explores sentiment patterns by scanning the input feed of semantic terms in only one direction. This direction concedes with the sequence of terms in the input feed.
2. We employ pre-trained word vectors and thus we were able to exploit the semantic relatedness that lied within those vectors.
3. We introduce the k-max pooling operation over the single-max pooling on top of a convolution layer.

4. We introduce a new attention layer that we place on top of a recurrent layer.
5. We use a number of additional datasets (nine in total) and several state-of-the-art benchmark methods to compare against our model's generalization.
6. We present a number of visualizations and analyses over the contribution of the k-max pooling operation in the improvement of generalization.
7. We present an error analysis on evaluation results and share with the community the predictive characteristics of our method.

All in all, this work, as illustrated in the experimental part (Sections 5, 6, 7, presents significant improvements in innovation components and classification performance.

2. Related Work

Convolutional Neural Networks. One of the pilot neural approaches that managed to infer the ratings of individual sentences using full-review ratings was introduced by the work presented in [6]. There a novel objective function was proposed for multi-instance learning. A key property in their method was a deep multi-instance convolutional structure that exploited embeddings for words, sentences and documents. Transfer learning and the significance of higher level embeddings was also highlighted in [15]. In general convolution neural networks have been adopted in many text and sentiment classification tasks [4, 16]. A milestone was achieved in [17]. There a successful demonstration of convolution networks on Natural Language Processing tasks resulted in adopting this architecture for text classification.

Recursive Neural Networks. In [18, 19, 11] a recursive neural tensor was employed to address the challenges of sentiment compositionality. Improved semantic vector spaces interacted with word vectors and higher-level node vectors in a tree-like form to produce sentence representations. Despite the supreme performance against neural bag-of-words models and other state of the art works in binary and fine-grained classification tasks, the model's performance suffered from parsing errors and limited capacity of the extracted patterns. This was more evident at the fine-grain task. A variant of the above mentioned methods proposed in [20] demonstrated a significantly better fine-grained patterns capacity, after improving the hierarchical representation of the input feed and the sentence's compositionality.

Pattern's Extraction. Later works proposed in [21, 3] addressed the above issue and introduced a deep convolution neural network which combined local pattern extraction and sentence compositionality. This latter attribute was achieved by the use of a k-max pooling operation that was applied after the convolution process. The k-max pooling operation resulted in producing a deep subgraph network which pruned the redundant patterns from the most useful ones. One of the disadvantages however was the high complexity of the model because of the successive convolutions. [5] tackled that issue and proved that even a single layer (multi-filter) convolution neural network can produce state of the art results in binary and fine-grained sentiment classification tasks, especially for sort text snippets that are met in micro-blog posts [22]. A significant asset that also played a key role in this generalization improvement was the use of pre-trained distributed word representations [23, 24, 25]. An asset that was also illustrated in [26] after experimenting on several sentence compositionality functions. One disadvantage however, is that the single max-pooling operation cannot grasp sentiment fluctuations that are quite abundant in long range opinion reviews. This is an observation that was also verified in the work of [27], where after applying a k-max pooling operation on a convolution structure it out-performed the single-max pooling, and also other neural network variants in sentiment classification tasks. Different in comparison to our implementation, the work presented in [28] first augments the input features by using a combination of a Bi-RNN network and an attention layer. Then, on top of this encoded output the authors apply sentiment classification via convolution layers. One disadvantage however, is that the extracted classification patterns after the convolution layers are impaired by the single-max pooling operation. Following the above remarks, in the experimental setup we use such rich opinionated datasets and demonstrate that this generalization weakness exists in the single max-pooling operation. In our method we adopt the convolutional structure that was utilized in [5] and we apply a k-max pooling operation [3] over the convolution layers. This operation aligns with the observation introduced in [26], that some key n-gram phrases are significant in sentiment prediction. In the same line of work with [29] they demonstrated that by partitioning a sentence by type can exploit the semantic relatedness between aspect-opinion terms and further improve the performance of sentence-level sentiment analysis. By adopting the above combination of [5, 3] processes in the convolutional layer we aim at extracting an enriched sentence embedding that will feed the recurrent steps of the proposed model.

Recurrent Neural Networks. RNN are sequential-like structures that are capable of grasping strong dependencies among the input features. The development of the LSTM architecture [30] designated the start of a new era in the exploration of sequential-like data where past structures [31] failed to generalize well [32, 33]. The main attribute in this architecture is the block (which is called 'memory cell') is able to preserve states over long sequences, and it also distinguishes inner actions from the outside world. This trait provides a powerful mechanism of choosing whether an incoming temporal information is important and should be taken into account and forwarded to the core of the memory cell or blocked and filtered out. A simplified version of this architecture was also used successfully for short text snippets [34]. A more sophisticated version of LSTM is the Hybrid Bi-Directional LSTM that captures semantic representations in both directions introduced in [35]. The attributes of this architecture were also employed in [36], where a k-max pooling operation applied on the Bi-RNN features. Despite the moderate results in several classification tasks, it was verified that this operation can extract the most significant patterns and improve performance. Several variants including linear and tree chain structures were also used for text and sentiment analysis tasks [12]. Their performance has been evaluated successfully on the exploitation of several input features. From the commonly used word embeddings, while also in sequences of higher-level phrase representation encodings [37].

One of the disadvantages of the above RNN methods, is their difficulty to train successfully over long sequences. In addition, simple recurrent neural networks tend to have a bias over the last sequences [9]. This gives the RNN excellent performance at language modelling, but it is sub-optimal for remembering the input sequences further back in a sentence. In our method we alleviate this issue by first introducing a sentence embedding over a single word embedding in an RNN input which significantly reduces the number of sequences. Secondly, we alleviate the bias problem by adopting a Bi-directional LSTM structure which exploits separately the forward and the backward semantic paths.

Transformers. Recent advances in neural networks [38] are based on sequence transduction models that employ an encoder-decoder configuration. Motivated by the attributes of this architecture, several works have used parts to implement domain specific tasks. In sentiment analysis the work of [39] used a multi-resource attention network for sentiment disambiguation of sentences. The method which scored remarkable results, asserted that the enrichment of the input features can significantly improve the generalization of a model in a task. In the same line variants of the transformer's architecture, like the ELMo [40], the GPT [41], the BERT [42] model have utilized in their implementation either the encoding or the decoding part to create a language model. Their successful evaluation in language understanding tasks has resulted in changing the landscape in many Natural Language Processing (NLP) tasks. Language understanding models encompass rich general semantic and syntactic information that can be exploited via transfer learning. These attributes while also their ability to be trained and fine-tuned in a completely unsupervised manner has made them particularly attractive in many downstream NLP tasks. As a result, transfer learning via the pre-trained language models has substituted the semantic relatedness of the pre-trained distributed word representations in many domain specific tasks. Their easy adaptation, has also contributed towards the adoption of such language modeling encoders in text classification and sentiment analysis tasks.

In our method we have employed, adapted and modified the attention mechanism presented in [38] to achieve two purposes. The first, to filter noise stemming from recurrent data and the second to improve generalization providing corpora or domain information during training at the optimization task. We also note here, that despite the rich linguistic features the pre-trained language models provide, this work addresses on an architecture that relies on pre-trained distributed word representations for the implementation of the domain specific tasks.

Aspect-Based Sentiment Analysis. At this point we also discuss how document level sentiment analysis relate with the Aspect Based Sentiment Analysis (ABSA) [43, 44, 45]. In ABSA the task is to co-extract aspect-opinion term pairs. Then, aspects polarity can be inferred by analysing the respective opinion-terms. Finally, a sentence polarity is obtained after summarizing the polarity scores of the opinion terms. Despite the fact that ABSA is considered a fine-grained opinion mining task, the necessity to identify and analyse both aspects-opinion term pairs per sentence to infer sentence polarity, limits the applicability of this method for document level sentiment analysis. Our method alleviates sentence by sentence sentiment polarity identification, exploring, extracting and exploiting sentiment classification patterns that rely solely on document level user's evaluations. To this end no extra sources like opinion or aspect terms are provided in the algorithm to implement the classification task.

3. Our Method (HolC)

Some principles of our method (HolC, Holistic Cumulative sentiment classification) were initially inspired by [46] and [47, 48]. The architectures in these models assume that an opinion consists of a set of sentences and each one can receive a prediction score over a number of discrete classes (i.e. *negative, neutral, positive*). The overall sentiment is calculated by summing all partial predictions. Similar works (for example, see [5, 3]) propose models where opinions, during analysis, are transformed into a large sequence of words. Prediction is achieved by applying machine learning classifiers that assign labels (positive, negative, neutral) to that sequences of words.

The main difference between the two groups of models lies in the way they evaluate a user's opinion. The first group formulates the classification problem as a series of individual predictions: one for every sentence in the group of sentences. The second, formulates the problem as a single prediction over a long phrase.

Our model design and implementation bridges this gap and advances the state-of-the art by adopting an attention based convolution and a recurrent feed forward structure supported by a classical (single layer perceptron) layer that better grasps the user's intention.

Overview. HolC is organized into three feed forward blocks. In the first *"sentence level"* block (see Figure 2(a)) we use a multi-filter convolution layer to transform word embeddings into sentence embeddings. In the second *"document level"* block (Figure 2(b)) we feed the sentence embeddings into a modified Bi-directional LSTM recurrent neural network. Then a set of specifically designed Attention layer(s) receives the B-LSTM output predictions and applies filtering and optimization. Next, a classical layer employs a number of optimized output predictions.

Finally, in the third block (*"prediction level"* - Figure 2(c)) we forward the partial predictions into an appropriate classifier that merges all partial predictions. Figure 2 illustrates the model architecture and the corresponding blocks. In the following sections we analyze all the above components in detail.

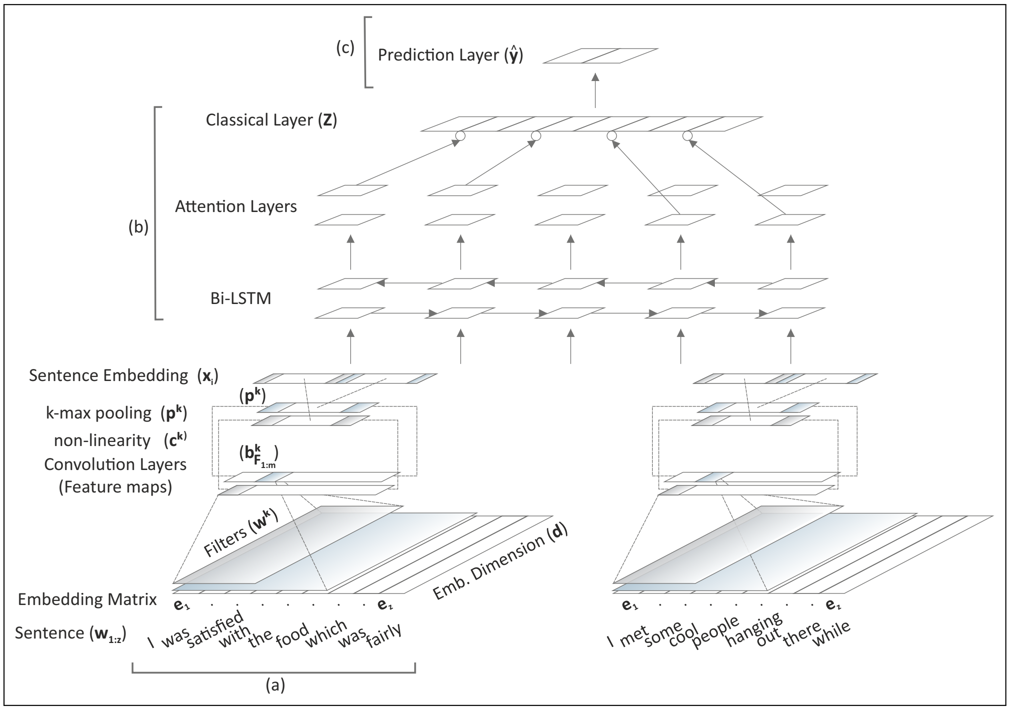
(c)
Prediction Layer ($\hat{\mathbf{y}}$)
Classical Layer ($\mathbf{Z}$)
(b)
Attention Layers
Bi-LSTM
Sentence Embedding ($\mathbf{x}_i$)
($\mathbf{p}^k$)
k-max pooling ($\mathbf{p}^k$)
non-linearity ($\mathbf{c}^k$)
($\mathbf{b}^k_{F_{1:m}}$)
Convolution Layers
(Feature maps)
Filters ($\mathbf{w}^k$)
Emb. Dimension ($\mathbf{d}$)
Embedding Matrix
$\mathbf{e}_1$ . . . . $\mathbf{e}_z$
Sentence ($\mathbf{w}_{1:z}$)
I was satisfied with the food which was fairly
(a)
$\mathbf{e}_1$ . . . . $\mathbf{e}_z$
I met some cool people hanging out there while

Figure 2: Model Architecture and The Feed Forward Neural Propagation.

3.1. Sentence Embeddings

As our architecture initially deals with raw words, we need to transform them into a suitable form for a neural network. A standard procedure in text analysis is first to create a vocabulary V from the corpus of opinions and a set of indices to map words into real valued vectors. Then a Look-up table is used to transform these indices into a form appropriate for a neural network. The NN layers that we use are the following:

Look-up Table Layer. A look-up table is created after we define the vocabulary size $|V|$ of the corpus and the embedding size d of the word vectors. This table $W_{(|V| \times d)}$ is a matrix of parameters to be learned. Each word is mapped to an index $i \in V$ and is embedded into a d-dimensional space and identified by the corresponding $W(i) = e_i$ index in the Look-up

Table. Thus, our network receives a sequence of words $\{w_1, w_2, \cdots, w_z\}$, where z is the sentence length that is initially transformed into a sequence of indices and then via a projection layer W into word embeddings of dimension d [2]. The matrix $[e_1, e_2, \cdots, e_z]$ of word vectors is the input of the first layer of this block. This process is repeated in parallel for all sentences τ of an opinion.

Convolution Layer. We begin in this layer with a tokenized sentence of length z which has previously been transformed into a matrix of word vectors of dimensionality d. Since rows represent word vectors it is reasonable to use convolutional filters with width equal to the dimensionality of these word vectors. Each filter consists of two dimensions the *height* i.e. the number of adjacent rows considered jointly and the *width* d. Both the dimensionality d and the height h comprise a set of $d \times h = w$ parameters.

A number of n filters convolve in the input matrix in parallel producing convolution layers of size $n \times b^k_{F_i} \times F_m$. Where $b^k_{F_i}$ a vector (feature map) which is produced after the convolution operation of each filter w^k on the input matrix and F_m the number of features maps per layer. Equation 1 presents how each element b^k_i in the feature map vector $b^k_{F_i}$ is calculated and Equation 2 the total feature map vector $b^k_{F_i}$.

$$b^k_i = w^k \cdot e_{i:i+h-1} \tag{1}$$

$$b^k_{F_i} = [b^k_1; \cdots; b^k_i; \cdots; b^k_z] \tag{2}$$

Each feature map $b^k_{F_i}$ produces an output vector of length z, where z is the sentence length. Finally, each filter k in a convolution layer produces an array of feature map vectors $b^k_{F_{1:m}}$ Equation 3 with size $b^k_{F_i} \times F_m$ and a number of tunable parameters w^k with $w^k = h^k \times d$.

$$b^k_{F_{1:m}} = [b^k_{F_1}; \cdots; b^k_{F_i} \cdots; b^k_{F_m}] \tag{3}$$

Non-Linear Layer. The convolution layer is a process that is appropriate for linear pattern extraction i.e. when there is linear correlation between classes and features. If the dimensional space is more complicated, a non-linear function should be employed. In our case we employ the matrix of feature map vectors $b^k_{F_{1:m}}$ of Equation 3 and apply a non-linear function f^{nl} in order to explore more complicated classification patterns. Equation 4 provides the arguments of the non-linear function. The result produces the vector c^k, with dimensions similar to $b^k_{F_{1:m}}$. Where β^k in Equation 4 is the bias parameter.

$$c^k = f^{nl}\left(b^k_{F_{1:m}}, \beta^k\right) \tag{4}$$

We apply k-max pooling over this array of vectors c^k (Figure 3) producing a matrix of size $k\text{-}max \times F_m$. Since different filters produce a matrix $k\text{-}max \times F_m$ separately, we refer to this matrix as partial sentence embedding. Equation 5 provides the calculation of the partial sentence embedding p^k.

$$p^k = k\text{-}max[c^k] \tag{5}$$

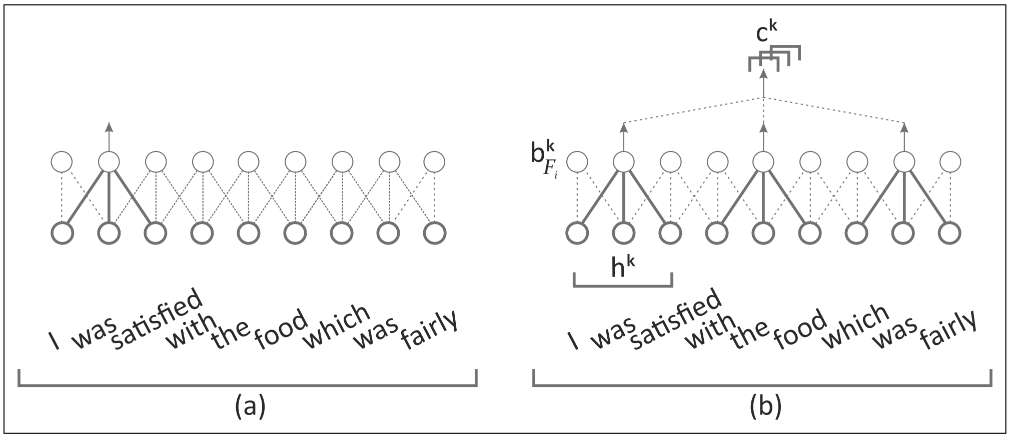
c^k
b^k_{F_i}
h^k
I was satisfied with the food which was fairly
I was satisfied with the food which was fairly
(a)
(b)

Figure 3: (a) The single-max pooling operation vs (b) the k-max pooling operation on the output vector $b^k_{F_i}$ of a convolution layer. The k-max operation, $k = 3$ in the example, can grasp better the sentiment fluctuations in a sentence to a single-max operation. Note that h_k stands for the filter height.

The convolution layers are responsible for grasping the most important sentiment patterns that exist in the corpus of opinions. This is achieved with the filter's parameters as described above and the k-max pooling layer we introduced, which enables more complicated patterns to be explored in the next layers.

$$x_i = [p^1; \cdots; p^k] \tag{6}$$

Finally we concatenate all p^k vectors to produce a sentence embedding x_i as shown at Equation 6. The vector x_i with size $n \times k\text{-}max \times F_m$ stands for the sentence embedding. All in all, since an opinion consists of τ sentences, we also have τ sentence embeddings $x = [x_1, \cdots, x_i, \cdots, x_\tau]$ that we forward to the next step of the neural network.

This block is responsible for converting a sequence of word embeddings into a sentence embedding or representation for every sentence in an opinion. By employing the sentence embedding which is produced after the above operations (Equations 4, 5, 6), we create a neural structure where on top, are the sentence embeddings vectors, consisting of k-max-pooled neurons and at the bottom are the convolution $n \times b^k_{F_{1:m}}$ output matrices. Recall that n is the number of filters.

3.2. Exploiting Semantic Embeddings

In this phase our network receives a sequence of sentences embeddings. We forward them to a modified Bi-directional structure that creates two independent recurrent sub-networks. Next, we present how these sub-networks are created and exploited. Before that, we present some basic concepts that are necessary for creating these sequences, namely the Bi-Directional Unit.

Bi-Directional Unit. Given an input sequence $x = (x_1, \cdots, x_\tau)$ a standard recurrent neural network (RNN) [49] computes the hidden vector sequence $h = (h_1, \cdots, h_\tau)$ and the output vector sequence $y = (y_1, \cdots, y_\tau)$, by iterating the following operations:

$$h_t = f(W_{xh} \cdot x_t + W_{hh} \cdot h_{t-1} + \beta_h) \tag{7}$$

$$y_t = W_{hy} \cdot h_t + \beta_y \tag{8}$$

where $(W_{xh}, W_{hh}, W_{hy}, \beta_h, \beta_y)$ are parameters and biases terms to be learned and f an appropriate non linear function.

In our method the sequence of vectors, the hidden h, and the output y are repeated in two recurrent layers. These are the forward and the backward layer and are implemented in a Bi-directional LSTM cell [30, 50]. For brevity we omit the rather extensive equations describing the LSTM network and we only refer to equations 9, 10, 11, 12 as the computations and parameters that have direct affect on our network architecture.

A standard BRNN [8] computes the forward hidden sequence $\overrightarrow{h}$, the backward hidden sequence $\overleftarrow{h}$ and the output sequences y by iterating the backward layer from t = T to 1, the forward layer from t = 1 to T and then updating the output layer:

$$\overrightarrow{h_t} = \sigma(W_{x\overrightarrow{h}} \cdot x_t + W_{\overrightarrow{h}\overrightarrow{h}} \cdot \overrightarrow{h}_{t-1} + \beta_{\overrightarrow{h}}) \tag{9}$$

$$\overleftarrow{h_t} = \sigma(W_{x\overleftarrow{h}} \cdot x_t + W_{\overleftarrow{h}\overleftarrow{h}} \cdot \overleftarrow{h}_{t-1} + \beta_{\overleftarrow{h}}) \tag{10}$$

$$\overrightarrow{y_t} = W_{\overrightarrow{h}y} \cdot \overrightarrow{h}_t + \overrightarrow{\beta}y \tag{11}$$

$$\overleftarrow{y_t} = W_{\overleftarrow{h}y} \cdot \overleftarrow{h}_t + \overleftarrow{\beta}y \tag{12}$$

However, as can be observed in Equations 11, 12 we exploit each layer's sequential output separately. Next, we construct a vector consisting of an array of predictions from these outputs which form a classical layer. We employ a hyper-parameter named *balancing factor* that immediately affects the classical layer's vector length. Figure 4 presents the Bi-Directional LSTM implementation in our network.

This part of the network is responsible for grasping sequential relations over the sentence embeddings. By combining the past (forward path) with the future (backward path) of the sentence embeddings in separate paths, we firstly counterbalance the bias term and secondly we improve the generalization of the network, a trait that we exploit appropriately at the next step of the network's information propagation.

Attention Layer. An attention layer is a computation mechanism that explores the most significant elements in an array of vectors. This, is achieved by exploiting the features among three vectors. The Queries Q, the keys K and the values V. The relation that binds these three vectors in the attention layer is given by the following equations 13, 14.

$$Q = y_t \times W_q,\ K = y_t \times W_k,\ V = y_t \times W_v \tag{13}$$

$$Att(y_t) = softmax(\frac{Q \times K^T}{\sqrt{h}}) \times V \tag{14}$$

Where $y_t = [\overleftarrow{y_t}; \overrightarrow{y_t}]$ is the attention input matrix. It is produced after the concatenation of the Bi-directional output of the equations 11, 12 of the previous layer, h is the attention depth with size $h = [\overleftarrow{h} ; \overrightarrow{h}]$ and W_q, W_k, W_v are learning parameters with size $h \times h$.

The Attention $Att(y_t)$ operation produces a weight average over the elements of the sequences part of an input matrix y_t. This input matrix has size $[m \times sequences \times h]$. Where m is the mini-batch size, consisting of m opinions. Normally, one would expect the sequences part to be the sentences embeddings. However, in our implementation, we swift the order of the input matrix in the form $[sequences \times m \times h]$. By applying this, revolving the input matrix on the attention layer, we provide additional info in the optimization process. That is we combine the corpus optimization with the document optimization. This process has been quite beneficial in the generalization of our model and we illustrate this in the experimental setup.

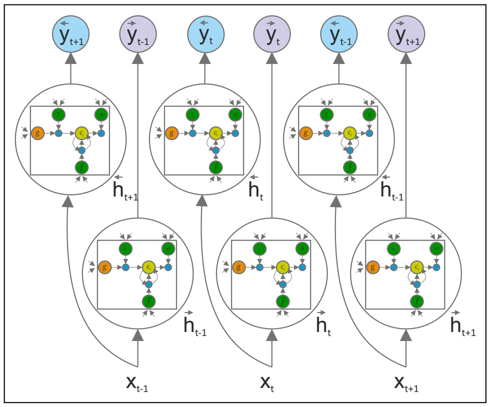
$\overleftarrow{y}_{t+1}$
$\overrightarrow{y}_{t-1}$
$\overleftarrow{y}_{t}$
$\overrightarrow{y}_{t}$
$\overleftarrow{y}_{t-1}$
$\overrightarrow{y}_{t+1}$
$\overleftarrow{h}_{t+1}$
$\overleftarrow{h}_{t}$
$\overleftarrow{h}_{t-1}$
$\overrightarrow{h}_{t-1}$
$\overrightarrow{h}_{t}$
$\overrightarrow{h}_{t+1}$
x_{t-1}
x_{t}
x_{t+1}

Figure 4: The BLSTM implementation and the separate output predictions.

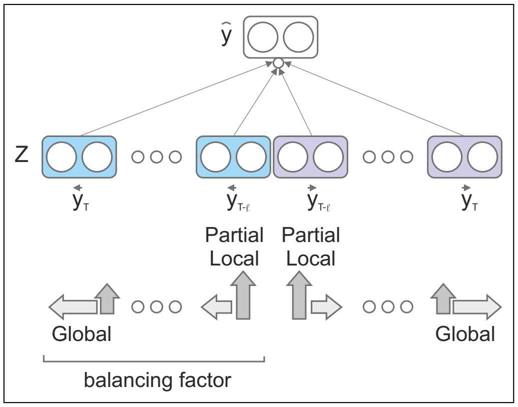

Figure 5: The Implementation of the Classical Layer and its role to the overall classification task.

Classical Layer. A number of output predictions from the RNN/Attention structure consists the input basis of a classical layer. We exploit a number of output predictions evenly from both directions. We refer to this number as *balancing factor* $\in [0, 1]$. Zero value yields the terminal output prediction each LSTM network outputs and one yields all output predictions respectively. If y_i is the prediction vector with size Δ that each RNN/Attention output layer produces (initially with size h and then after a linear transformation into size Δ) and $l \in [1, \tau]$ the equivalent length of the *balancing factor*, then a number of Z neurons produce a vector with size $2 \times \Delta \times l$ constitute the input vector basis of the classical layer. Next, we exploit these Z

values in Equation 15, by connecting them with the unregularized prediction layer of the network $\hat{y}$. Equation 16 provides that connection.

$$Z = [\overleftarrow{y}_{\tau}; \cdots ; \overleftarrow{y}_{\tau - l}; \overrightarrow{y}_{\tau - l}; \cdots ; \overrightarrow{y}_{\tau}] \tag{15}$$

$$\hat{y} = Z \cdot w_w^T + \beta_w \tag{16}$$

Where w_w^T with size $Z \times \Delta$ and β_w with size Δ in Equation 16 are the parameters and bias terms of the classical layer to be learned. Equation 16 calculates the final vector $\hat{y} \in \Delta$ that is forwarded next for normalization and prediction.

In this block we materialize two operations. First, we explore the sequential dependency of sentences provided via the sentence embeddings. Second, via the classical layer and the *balancing factor* we grasp both *"local"* (single sentences) and *"global"* (multiple sentences) predictions all together. Figure 5 depicts how the classical layer is implemented and portrays its role at the overall classification task. Each output prediction layer ($\overleftarrow{y}_t$, $\overrightarrow{y}_t$) encompasses two parts of information. One local and one global. As we move on to the terminal states, global information increases, and local diminishes.

3.3. Network Training

Given an opinion X_i with sentences $X_i = \{s_i^1, ..., s_i^{\tau}\}$, the network with parameter set θ computes a score $\hat{y}_{\delta}$ for each label $\delta \in \Delta$ in the prediction vector $\hat{y}$. In order to transform these vector's values into a conditional probability distribution, we apply the soft-max [51] operation over the scores of all tags $\delta \in \Delta$.

The result is a probability distribution over the Δ possible labels which in our case represent the prediction score of each sentiment label. Normally the prediction of the system is the label with the largest value. Next, we employ this vector of probabilities Δ and calculate the negative log likelihood (cross entropy criterion) to use it next for network training (see Equation 17).

When training deep neural networks it has been found advantageous to select mini-batches of size $m \in N$ that are selected randomly from the whole training dataset, rather than using utterances as batches [52]. Thus our network is trained over a batch of random set of examples m. Eventually every training iteration produces a loss or error J that we calculate in Equation

17.

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} y_i \; log \; p(\Delta|X_i, \theta) + \frac{\lambda}{2}\| \; \theta \; \|_2^2 \tag{17}$$

Where y_i in equation 17 are user's sentiment assignments in one hot vector form, and λ is the L_2 regularization term [53] that we set during the learning phase. Finally, the error $J(\theta)$ is a task that is undertaken by an optimization algorithm which implements the training phase of the model.

4. Hyper-parameters and Training Details

Here we present the details and the hyper-parameter settings of our model.

4.1. Preprocessing & Datasets

The pre-processing phase in our model consists of removing all opinion's delimiters (i.e:, ?, !, $\cdots$), apart from the full-stop which we use as a sentence separator.

We use either the average *(Avg)* or the maximum *(Max)* value of sentences per opinion to set-up the dimension of our model (see Table 1). Thus, the Avg/Max number of sentences per opinion frames our model's feeding input. For the corresponding words per sentence we always use the max number. We refer to this preprocessing parameter as *Input Size* (see Table 1). For alleviating the variable length of sentences, we apply padding with zeros over the remaining sequences of a sentence until the Avg/Max window length.

Reproducibility Note. All source code that is required to run the following experiments is available at the following link: `https://github.com/unic-ailab/Holistic-Cumulative`

Datasets. For the experiments, we focus on the following corpora.

- SST: Stanford Sentiment Treebank[3] - Movie reviews with one sentence per review provided with fine-grained labels.

[3] `https://nlp.stanford.edu/sentiment/`

- SST-bin: The SST dataset transformed into binary. The instances with the neutral labels were removed.
- YELP: A subset of the YELP reviews dataset[4].
- YELP-bin: The equivalent binary YELP dataset. The examples of the neutral labels were removed.
- MR: Movie reviews with one opinion per review. Classification involves detecting positive/negative reviews[5].
- SUBJ: A dataset of 5k subjective and 5k objective sentences[5].
- SEMEVAL: A collection of twitter messages[6]. We refer to the 2016 dataset and Subtask B: Tweet classification according to a two-point scale (positive/negative).
- TREC: A Question dataset which involves classifying a question into 6 question types[7]. This dataset is used in order check the performance of our methods in a general text classification task.
- DRANZIERA: A subset of a multi-domain dataset consisting of 200k positive/negative reviews[8].

Table 1 presents the basic characteristics of each dataset along with prepossessing settings. For the DRANZIERA dataset the basic characteristics along with the evaluation results were presented in Table 6 .

4.2. Hyper-parameters and Training Details

Table 2 presents the details of the hyper-parameters of all models. HolC stands for the proposed approach (Holistic Cumulative). The competitive algorithms are the BLSTM and the CNN which we implemented, while we also include other state-of-the-art benchmarks. In order to provide a fair

[4] `https://www.yelp.com/dataset`

[5] `http://www.cs.cornell.edu/people/pabo/movie-review-data/`

[6] `https://github.com/seirasto/twitter_download`

[7] `http://cogcomp.org/Data/QA/QC/`

[8] `http://www.maurodragoni.com/research/opinionmining/dranziera/protocol.php)`

Table 1: Datasets characteristics and Preprocessing settings

	MR	SEMEVAL	SUBJ	YELP-bin	SST-bin	YELP	SST	TREC
Opinions	10662	16801	10000	8538	9604	9999	11844	5952
Sentences / Opinion (Max)	6	8	5	63	4	80	5	6
Sentences / Opinion (Avg)	1	1	1	7	1	7	1	1
Words / Sentence	52	37	112	132	52	132	52	34
Vocabulary	18121	19057	22854	27559	15978	29584	17604	8286
Input Size	Max	Max	Max	Avg	Max	Avg	Max	Max
Train/Test	5-fold	pre-set	5-fold	5-fold	pre-set	5-fold	pre-set	pre-set

comparison among the alternative models and the HolC, all neural hyperparameters were setup jointly. HolC is the model as described above consisting of the blocks (a), (b), (c) in Figure 2. BLSTM is a Bi-directional Long Sort Term Memory Model presented in [35]. For the implementation of all RNNs, we employed the LSTM version presented in [10]. CNN is the Convolution Neural Network model presented in [5]. CNN-static is our implementation of the method presented in [5]. The pre-trained word vectors remained static during model training.

All neural models initiated their word vector representations utilizing the fasttext[9], two million word vectors trained with subword information on Common Crawl [25] that we maintained static during training. The hyperparameters for our model are tuned during the training and the development set for each task.

For all classification tasks, word representations remain static, all other model parameters are tuned during training using the Adam [54] optimization algorithm with a learning rate of 1.7×10^{-3} using also exponential decay during the training process. Additionally model parameters are regularized with a per-minibatch L_2 regularization strength of 10^{-6} for the convolution and 10^{-5} for the rest model's parameters. Additionally HolC was regularized [55], with an initial dropout rate of 0.5 that was dynamically modified to higher values when over-fit was identified between the training & development datasets. For the Attention part of our method we used 2 layers. The rest of the neural models CNN, BLSTM were simply regularized with a steady rate of 0.5 dropout and L_2 10^{-5}. All datasets were split in train/test parts following the norm 80% training 20% testing, unless stated otherwise. For more information about the train/test splits, please see Table 1. 5-fold

[9] https://fasttext.cc/docs/en/english-vectors.html

Table 2: Model's Hyper-parameters

	HolC	BLSTM	CNN
Embedding dimension	300	300	300
Hidden size	125	125	-
Feature's maps	128	-	128
Balancing Factor	[0-1][a]	-	-
Batch size	32	32	32
Num epochs	45	45	45
Filter Heights	4[b],3,2	-	3,2
Top-k	[1-10][c]	-	1

[a] We refer to the "balancing factor" value the model performed the best generalization.

[b] SST, SST-*bin*, SEMEVAL and SUBJ benchmark datasets were fine-tuned with this extra filter.

[c] We refer to the top-k values the model performed the best generalization.

means we split the dataset following the norm 80% training 20% testing in 5 iterations. Pre-set means we employed the predefined train/test partitions of the respective dataset. 10% from every training dataset was reserved for development. Each experiment was iterated five times in a 5-fold setting and the accuracy of each model was based on the respective average value. After every iteration the model's parameters were fine-tuned from the beginning.

5. Results

In this section we evaluate HolC's performance on two types of problems: Sentiment classification and Question type classification.

5.1. Sentiment Classification

In this subsection we evaluate our model's performance on the sentiment analysis task. We employ the datasets of Table 1 and set-up all models with the hyper-parameters values of Table 2. All experimental results refer to the accuracy metric which was measured on the corresponding test-sets after training the models. From now on we will refer to accuracy as the generalization of a model. We note that the comparisons that follow-up refer solely to models that employed pre-trained word embeddings as extra features to boost their results. Evaluations of pre-trained language models are provided at the end of this section.

In Table 3, we notice that the proposed model, HolC, present better generalization over the implemented algorithms (*) in all cases (7/7). The reader can notice that at the SUBJ dataset none of the alternative algorithms we implemented CNN(*), BLSTM(*) was able to generalize well.

This could possibly be due to the cross-entropy criterion that does not manage to affect HolC. However, note that all models (including HolC) were trained with the same settings. In Table 3 we also present the maximum values achieved by our method within the 5-fold setting (HolC$_{max}$).

What we observe is that performance is even better when parameters are optimized (especially for the pre-set datasets of Table 1 where splits remain the same during iterations). Moreover, these results outperform the competitive methods like [60, 5, 26, 15, 64, 65, 66, 67] (in dataset SUBJ), methods [60, 5, 26, 62, 63, 15, 65, 66, 67] (in dataset MR), methods [22, 34, 69, 71, 72, 73] (in dataset SEMEVAL). In methods [61, 3, 5, 11, 37, 26, 66, 68] (in dataset SST) and in some cases like [60, 11, 26, 68] (in dataset SST-bin) HolC, outperformed these benchmark methods.

At Table 3 we also note the contribution of the Attention mechanism we introduced earlier in our model implementation. We observe that the attention favoured all datasets, however, this contribution was more beneficial at the fine-grained and less at the binary tasks. We attribute this to the fact that in the fine-grained datasets the Attention layer was more beneficial because the complexity of the optimization problem had to deal with more classes. Moreover, the fine-grained datasets require the learning of more complex patterns, their best scores achieved at greater top-k values with respect to the binary datasets (see Figures 6, 7). Consequently, where we had more patterns and more optimization paths we observed greater contribution of the Attention mechanism (datasets YELP, SST).

Additionally one more asset of the HolC model, is its ability to generalize remarkably well when the dataset is rich in sentences throughout its training corpus. This is evident in the YELP dataset (Table 1) where there are 7 sentences per opinion by average, which is much larger than the other datasets.

Here, we present the evaluation results of some pre-trained language models that were employed and fine-tuned for the benchmark datasets we also present at Table 3. More specifically for the SST dataset, the BERT$_{large}$ [42, 56] model performed accuracy 55.50%. For the SST-*bin*, the BERT$_{large}$ [42,

Table 3: Accuracy of our method with optimized parameters (HolC$_{max}$) for all datasets vs HolC with the common set of parameters (see Table 2). The star (*) symbol means that the method is implemented by our research team based on the details provided by the corresponding papers (also mentioned in this Table). Where there is Underline, it indicates that the result is statistically significant at the level of 0.05[a]. Boldfaced values indicate the higher results.

	SUBJ	YELP-bin	SEMEVAL	SST-bin	MR	YELP	SST
HolC$_{max}$	$\underline{94.80\%}$	**91.16%**	$\underline{84.96\%}$	86.27%	$\underline{84.01\%}$	**52.75%**	$\underline{49.28\%}$
HolC	94.11%	**90.00%**	$\underline{84.64\%}$	85.48%	79.78%	**50.75%**	48.62%
HolC w/o Att	93.54%	89.54%	84.10%	84.51%	79.69%	49.46%	47.52%
CNN-static [5]$^{(*)}$	50.51%	77.82%	71.28%	83.51%	78.30%	31.19%	43.28%
BLSTM$^{(*)}$	49.87%	75.57%	70.19%	84.34%	77.48%	31.03%	43.52%
BiLSTM-Max [60]	92.40%	-	-	84.60%	81.10%	-	-
DAN [61]	-	-	-	86.30%	-	-	47.70%
DCNN [3]	-	-	-	86.80%	-	-	48.50%
CNN [5]	93.40%	-	-	87.20%	81.50%		48.00%
RecNTN [11]	-	-	-	85.40%	-	-	45.70%
CT-LSTM [12]	-	-	-	88.00%	-	-	51.00%
C-LSTM [37]	-	-	-	87.80%	-	-	49.20%
SWEM-concat [26]	93.00%	-	-	84.30%	78.20%	-	46.10%
RNN-Capsule [62]	-	-	-	-	83.80%	-	49.30%
MEAN [39]	-	-	-	-	**84.50%**	-	**51.40%**
AdaSent [63]	**95.50%**	-	-	-	83.10%	-	-
USE [15]	93.90%	-	-	87.21%	81.59%	-	-
Fast Dropout [64]	93.60%	-	-	-	-	-	-
SDAE [65]	90.80%	-	-	-	74.60%	-	-
GRU-RNN [66]	91.85%	-	-	-	78.26%	-	45.02%
Capsule-B [67]	93.80%	-	-	86.80%	82.30%	-	-
Emo2Vec [68]	-	-	-	82.30%	-	-	43.60%
BiLSTM-CRF & CNN [29]	-	-	-	**88.30%**	82.30%	-	48.50%
SwissCheese [22]	-	-	82.00%	-	-	-	-
CUFE [34]	-	-	83.40%	-	-	-	-
ECNU [69]	-	-	84.30%	-	-	-	-
UNIMELB [70]	-	-	**87.00%**	-	-	-	-
Thecerealkiller [71]	-	-	82.30%	-	-	-	-
TwiSE [72]	-	-	82.60%	-	-	-	-
Finki [73]	-	-	84.80%	-	-	-	-

[a] Language model evaluations [42, 56, 57, 58, 59] that are referenced at the end of this subsection are also included (where applicable) in this result.

56], the XLNet [57], the RoBERTa$_{large}$ [58] and the SMART-RoBERTa$_{large}$ [59] have similarly scored 94.90%, 97.00%, 96.90%, and 97.50%. Finally, at the SUBJ dataset, the SMART-RoBERTa$_{large}$ [59] method scored 97.10%.

Overall, the results of this section at the task of sentiment analysis highlight the advantages of the proposed model against the competitive benchmarks.

5.2. Question Type Classification

The TREC questions dataset involves six different question types, e.g. whether the question is about a location, about a person or about some numeric information [74]. The training dataset consists of 5452 labelled questions for training and another 500 for testing. Table 4 presents the average results for all methods.

Table 4: Accuracy (Average and Best value) of the six-way question classification on the TREC dataset. The star (*) symbol means that the method is implemented by our research team based on the details provided by the corresponding papers (also mentioned in this Table). Boldfaced values indicate the highest score in each dataset (column). Underlined values indicate that the result is statistically significant at the level of 0.05.

Method	Acc	Best Acc
HolC	**98.64%**	**99.00%**
HolC w/o Att	98.48%	98.80%
CNN-static [5](*)	98.60%	98.80%
BLSTM(*)	97.84%	98.80%
CNN [5]	93.60%	-
AdaSent [63]	92.40%	-
BiLSTM-Max [60]	88.20%	-
DCNN [3]	93.00%	-
USE [15]	98.07%	-
SDAE [65]	78.40%	
GRU-RNN [66]	93.00%	-
Capsule-B [67]	92.80%	-
SWEM-aver [26]	92.20%	-

The results at Table 4 indicate that the proposed model (HolC) average values outperform all neural competitors.

On average all models performed well on the TREC dataset and the best observed values prove that the HolC model is more accurate with other state-of-the-art models at a generic text classification task. This provides evidence that HolC, although built for sentiment analysis, it is not limited to this task.

6. The Contribution of the k-max Pooling Operation

This section studies the contribution of the k-max pooling operation, introduced at the convolution layer, towards the model's accuracy. Figure 6 presents the accuracy distributions for the fine-grained classification datasets, while Figure 7 the respective accuracy distributions for the binary classification datasets.

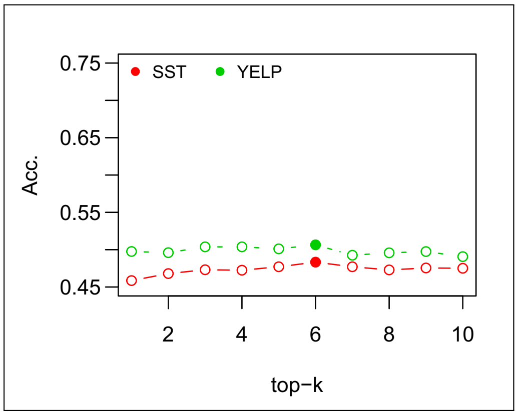

Figure 6: HolC (with Attention) Accuracy in fine- grained datasets for various values of k.

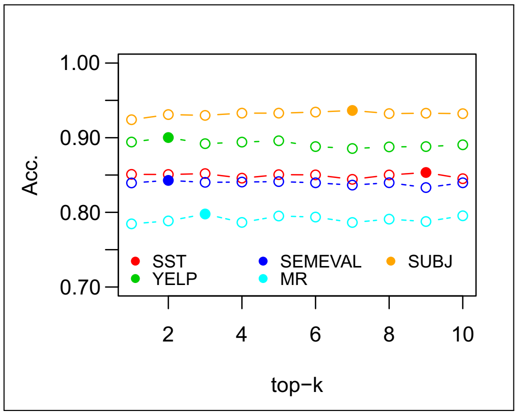

Figure 7: HolC (with Attention) Accuracy in Binary datasets for various values of k.

Comparing the two figures, it is clear that the k-max pooling operation contributed more in the fine-grained than in the binary classification task. We also observe that in all cases (but for TREC, which is not a sentiment dataset, the top-k value was one), the best accuracy obtained for k value greater to one.

Comparing to top-k=1 for HolC vs optimum top-k HolC, we obtained accuracy improvements like 0.89%, and 2.76% in datasets YELP, SST and 0.39%, 0.70%, 1.70%, 1.32%, 0.58% in datasets SST-bin, SEMEVAL, SUBJ, MR, YELP-bin. For the fine-grained tasks (see Figure 6) the best accuracy was achieved at $k = 6$ for datasets SST, YELP, while for the binary Figure 7 are $k = 2$ for YELP, SEMEVAL, $k = 3$ for MR, $k = 7$ for SUBJ and $k = 9$ for the SST dataset.

What is also significant to note over the accuracy results in Figures 6, 7 is that after a certain top-k optimum value the accuracy degrades. This is an indication of two things. First, every dataset is exploited optimally at a certain top-k value which corresponds to the optimum number of sentiment patterns. Second, more top-k values are redundant and as a result they obfuscate the optimization task. We also note that this effect is more evident at the fine-grained datasets.

6.1. Holistic vs Cumulative Content

At this section we explore how the dynamic characteristics of the classical layer (see related paragraph at section 3.2), and more specifically the *balancing factor* parameter can help us understand the degree the users of a specific corpus assign their rating in a cumulative or a holistic way. Adjusting the parameter will help the method improve performance.

In this set of experiments, we use the hyper-parameter values stated in Table 2. We change the ***B**alancing **F**actor* (**BF**) hyper-parameter value from zero to one with a step of 0.25. HolC with value 0 in this parameter will perform well if the users follow the cumulative strategy of rating (considering and weighting the advantages and disadvantages of the object under review). On the other hand, a HolC model with value 1 in *balancing factor* indicates that the users of that specific source rate the objects in a holistic way (by considering the big picture). To illustrate this concept with an example we present the following opinion snippet from the YELP-bin dataset in Figure 8.

An analytic user would provide neutral in case of fine-grained sentiment assignment or negative in case of binary sentiment assignment. The cumulative strategy of our model predicted negative for that review, whereas the holistic strategy predicted positive, which was in line with the reviewer's sentiment assignment.

Table 5: Holistic/Cumulative content identification over a set of different datasets & *balancing factor* (**BF**) size values (average accuracy of repeated experiments). Boldfaced values indicate the highest results. Underlined values indicate the second best results.

BF	SUBJ	YELP-bin	SEMEVAL	SST-bin	MR	YELP	SST	TREC
0	**94.11%**	89.67%	**84.64%**	**85.48%**	**79.78%**	**50.75%**	**48.62%**	98.60%
0.25	93.38%	89.66%	84.09%	84.95%	79.47%	50.19%	47.13%	98.44%
0.5	92.23%	89.80%	84.28%	84.63%	79.03%	49.16%	47.65%	98.52%
0.75	93.28%	89.52%	84.19%	85.10%	78.73%	50.65%	47.78%	**98.64%**
1.00	93.50%	**90.00%**	84.31%	84.96%	79.22%	49.85%	47.04%	98.60%

> “I’d love to give these guys a better review, because they were very friendly and professional (+). However, I was very dissapointed with the service I received (−). I’ll type the one sarcastic comment that I can’t get out of my head and then I’ll get to the more substantial feedback (−). I thought a detail was supposed to be a little more DETAILED (−). Here are the positives, because I want to give credit where credit is due (+). The guys come to you and are very polite (+). They were a little delayed, but called to let me know which I appreciated (±).”

Figure 8: Example of Holistic vs Cumulative Classification strategy on an Opinion snippet from the YELP dataset

The results of the experiment are presented in Table 5. A few observations are the following. Most sentiment datasets (but for YELP-bin) present best accuracy at value 0. This reveals that the content on these datasets is mainly cumulative. However, one easily notices that the second or third best scores in these datasets (underlined values) are either 0.75 or 1.00, which means that the reviewing process is holistic. The reverse observation is also valid for the primarily holistic YELP-bin which presents the third best score at value 0. Consequently, we may safely infer that in opinionated datasets lie both, the holistic and the cumulative sentiment assignment in user behaviour.

Another important conclusion of this section is the results of the TREC dataset. What can be observed there is that the parameter we introduced earlier, the *balancing factor* does not really affect the results of the classification task. This is rather interesting since it confirms the utility of this parameter for sentiment analysis tasks. TREC, as described earlier, is a question classification dataset and therefore the content is not opinionated. These results confirm that the parameter, *balancing factor*, is suitable for sentiment analysis tasks but does not obfuscate the task when the content is not opinionated.

Moreover, the extensions of the *balancing factor* could also be important for research that is related to psychology and relevant fields that study human behaviour [75].

7. Multi-Domain Sentiment Classification

In this section we test the performance of our method on a multi-domain sentiment classification task. The challenge in this setting is to assess the ability of our method to adapt successfully in different domains. Usually a model that performs well in one domain, it performs poorly or sub-optimal when the number or the variability of the domains increase. Here, we evaluate HolC on a dataset of user's reviews derived from several domains.

Table 6: Statistics & Evaluation Results of HolC on the DRANZIERA Multi-Domain dataset. Boldfaced values indicate the highest results.

Domain	Opinions	Sentences / Opinion	Words / Sentence	Vocabulary	HolC	NeuroSent [76]
Amazon Instant Video	10k	6	132	35533	**85.76%**	80.17%
Automotive	10k	4	256	18604	85.32%	**85.37%**
Baby	10k	4	272	16179	**87.09%**	85.18%
Beauty	10k	4	246	17898	**86.77%**	85.50%
Books	10k	6	346	38926	**84.05%**	79.66%
Clothing Accessories	10k	4	349	10497	**95.24%**	86.96%
Electronics	10k	5	203	22775	86.13%	**86.41%**
Health	10k	4	184	18450	**86.40%**	86.11%
Home Kitchen	10k	5	309	19578	**88.35%**	86.86%
Movies TV	10k	6	233	36138	**86.57%**	80.90%
Music	10k	6	132	34894	**86.95%**	80.83%
Office Products	10k	5	231	19928	86.17%	**87.30%**
Patio	10k	5	179	19933	**86.21%**	85.64%
Pet Supplies	10k	4	284	18318	**86.82%**	83.61%
Shoes	10k	4	309	8871	**96.21%**	86.55%
Software	10k	6	237	25173	**85.14%**	84.79%
Sports Outdoors	10k	4	309	18160	**89.51%**	86.69%
Tools Home Improvement	10k	5	421	20536	84.78%	**85.18%**
Toys Games	10k	4	343	19405	**87.32%**	86.24%
Video Games	10k	6	216	29347	**84.53%**	82.06%
Average					**87.27%**	84.60%

We consider suitable for this purpose the DRANZIERA dataset [77]. This is a multi-domain dataset composed of one million reviews crawled from the Amazon web site. For the evaluation of HolC, we employed a single data partition from each domain. Each partition consists of 10k reviews which are split equally in 5k positive and 5k negative reviews respectively. According to the evaluation settings proposed in [77], HolC evaluated all domains following the “Closed” setting in a 5-fold cross validation. The Hyper-parameters for all evaluations were set according to Table 6, except for the *k-max* and the *BF* value. These were set to 3 and 0 respectively, after considering the best accuracies, based on these hyper-parameters for most of the binary datasets. (See also Figure 7 for the top-k, and Table 5 for the *Balancing Factor*, where best scores achieved in these datasets.)

Table 6 presents the domains of the DRANZIERA dataset, while also input feed statistics and the evaluation results. What we observe is that HolC performed well in all domains. Moreover, in comparison to the baselines presented in [76], HolC outperformed these benchmark methods in most cases. The average score was also significantly higher. Consequently, we infer that HolC is a model and architecture that adapts successfully on a multi-domain setting. This is a significant asset which additionally provides more ways to utilize the HolC model for general purpose solutions.

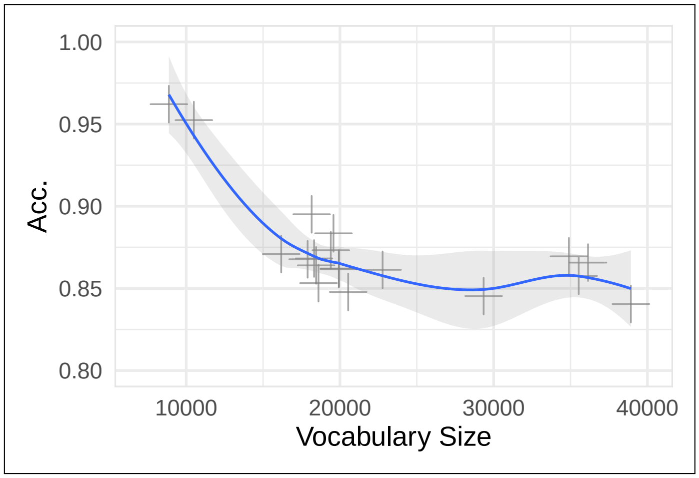

Figure 9: Accuracy vs Vocabulary Size in the evaluation of the DRANZIERA dataset.

Here we also discuss some characteristics of HolC that we believe they will provide insights about the architecture and the features it exploits. More specifically we note the vocabulary size from each domain and the respective accuracy scores from Table 6 and analyze them. Figure 9 presents the result of this analysis. Easily one observes that as the vocabulary size increases the accuracy scores degrade. We attribute this behavior to two reasons. We address the first to the sentiment disambiguation problem in opinion sentences which we hypothesize gets more evident as the vocabulary size increases. According to this, text snippets that include similar terms, may be encountered in different polarity contexts. As a result the polarity identification in the respective sentences becomes uncertain. The second relates to the features the architecture exploits. As it was explained in the description of the method, the bottom part extracts k-max n-gram features (mostly bi-grams and three-grams) from the convolution filters. These features are tailored to grasping successful sentiment fluctuations in sentences, however, they are not able to encode higher level content understanding. This results in sup-optimal performance when sentiment disambiguation or more complicated content cases are encountered. This is an issue that we also discuss in more detail in the error analysis section.

8. Error Analysis

This section presents an error analysis of our method. It aspires to shed light on the strong while also the weak predictive attributes of the proposed architecture and share the outcome of this investigation with the community. We rely on the scores that we obtained after we processed the fine-tuned models on the respective evaluation datasets. We opt for a representative dataset that we believe encapsulates both binary and fine-grained classification characteristics. We select the widely known SST (fine-grained) dataset and the respective test-set prediction scores. We analyse these results and calculate the normalized confusion matrix as shown in Figure 10.

We also provide some statistics of the train and the test partitions of the SST (fine-grained) dataset as shown in Figure 11. Please note, that these partitions are predefined and common to all methods that are evaluated on this dataset. First, we discuss the dataset characteristics in Figure 11. This Figure presents the distribution of the reviews per class in the train and the test sets respectively. Comparing the two distributions, we notice that the train set distribution presents some degree of imbalance. This is because the greater mass of reviews is gathered around the negative, the neutral and the positive class (labels $1, 2, 3$) respectively. As regards the rest, the very negative class (label 0) and the very positive class (label 4), the number of reviews fall relatively behind from the negative (label 1) and the positive (label 3) respectively. In the test set however, the classes are represented more uniformly. Evaluating the above remarks, we infer that imbalance exists only in the train set in two cases. The first is in the negative vs very negative class

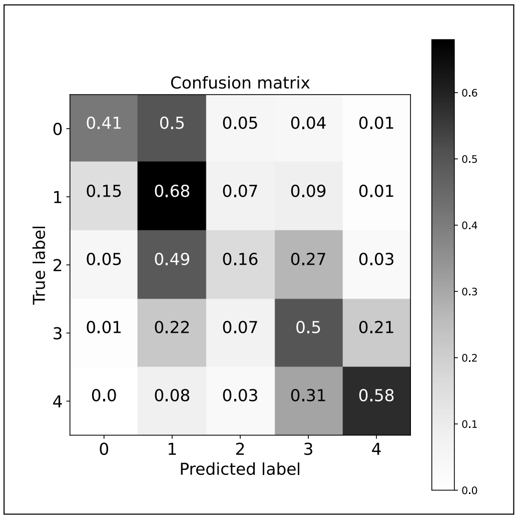

Figure 10: The normalized confusion matrix derived after the analysis of the evaluation results at the SST dataset.

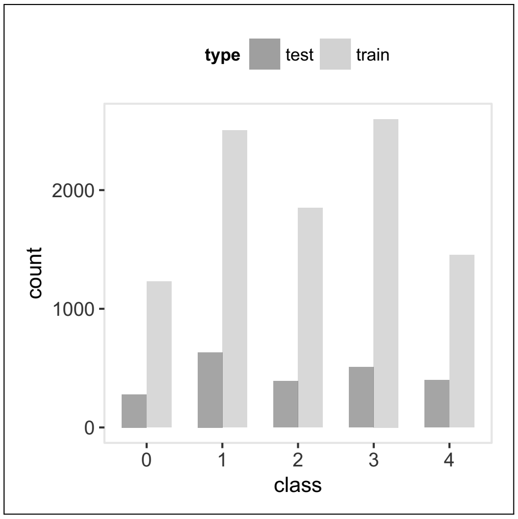

Figure 11: Distribution of the train & test set reviews per class in the SST (fine-grained) dataset. Horizontal axis depict the classes, vertical axis the number of reviews per class.

case (label 1 *vs* 0) and the second is in the positive vs very positive class case (label 3 *vs* 4) respectively. Despite this imbalance, we don't encounter any prediction inconsistencies in the scores of the confusion matrix that reveal optimizations issues or under-representation in the training examples. We base this inference on the observation that the greater mass of accurate predictions (gray shaded values) are concentrated around the diagonal.

Now, we comment on the results we obtained from the confusion matrix in Figure 10. Easily one observes that the neutral class performed the lowest prediction score. We attribute this prediction behavior to the following reasons. First, reviews that fall in this class, we believe include linguistic terms which partially overlap with terms they are also used in other classes such as the positive or the negative. This can also be verified from the predictions scores 0.49 as negative (Predicted label 1) and 0.27 as positive (Predicted label 3) for reviews that belonged in the neutral class (True label 2). Second, we support the hypothesis that accurate prediction of the neutral class reviews, requires neural features that encompass higher level content understanding. The pre-trained language encodings that were introduced earlier encompass such strong syntactic and semantic relations among the sentence terms. This is an asset, that methods which employ pre-trained language models are able to overcome this prediction bottleneck successfully and perform state-of-the-art results.

On the other hand, the sentiment classification patterns our method extracts are able to predict successfully reviews in the positive and the negative classification areas. As an indicative example we point to the set of (gray shaded) prediction values above and below the diagonal. These values in the binary classification case they would normally be merged in the diagonal of the confusion matrix, indicating successful prediction scores.

All in all in this section we provided evidence of the prediction performance of our method. HolC is able to adapt successfully to the majority of positive and negative classification examples, however, the extracted patterns provide poor performance in identifying the neutral cases. Additionally, we discussed the significance of the input features and how these can affect the ability of the model to generalize well in all classification cases.

9. Conclusions and Future Work

In this paper we proposed an improved version of the novel hybrid Convolution - Recurrent Neural framework presented in [14]. The novelties of the method proposed in this paper are the following: 1) The introduction of a sentence embedding via a Convolution Neural Network 2) a bi-directional recurrent neural network for encoding semantic content sequentially, 3) a classical layer that exploits both local and global information 4) a hyper-parameter that balances mixed content motifs named *Balancing Factor*, initially *Output Window Size*. In addition, this model introduces 5) an improved convolution operation that better exploits the input information, 6) a k-max-pooling operation over the single max-pooling after the convolution layer, 7) an improved design of the attention layer capable of improving the generalization task and, 8) the utilization of pre-trained word vectors over the randomly initialized ones. We experimented on a set of different datasets in binary and multi-class classification and studied their performance. On average the performance of the model improved its predecessor [14] by 15% values of accuracy. It also outperformed in 4 and succeeded statistically significant results in other 4 against the competitive methods (see Tables 3, 4, 6). Overall, in 8 out of 9 cases. One of the major advantages of our model, HolC, is the introduction of a hyper-parameter that can tune the classification model and perform better in situations where the users have holistic or cumulative behaviour while they rate products, services etc. Also the k-max pooling operation proved to be an essential asset towards the generalization improvement, especially in the sentiment analysis of fine-grained tasks and in datasets with rich content (i.e. YELP).

Pre-trained word vectors were also used in our method, however, it was the building blocks and layers we introduced, which enabled the exploitation of the global semantic relatedness that existed in these word vectors. The Attention layer, part of our model was in line to exploit and augment this semantic relatedness. Newer methods like pre-trained transformer based models on unlabeled data provide richer inner word semantic relatedness and have advanced the state of the art in many tasks. In future work we aspire to make use of such models and experiment on sentiment analysis tasks.

Acknowledgments. This project has received funding from the European Union's Horizon 2020 research and innovation programs under grant agreements No 825171 and No 101017558.

References

[1] S. Katz, Estimation of probabilities from sparse data for the language model component of a speech recognizer, IEEE Transactions on Acoustics, Speech, and Signal Processing 35 (3) (1987) 400–401.

[2] Y. Bengio, R. Ducharme, P. Vincent, C. Janvin, A neural probabilistic language model, J. Mach. Learn. Res. 3 (2003) 1137–1155.

[3] N. Kalchbrenner, E. Grefenstette, P. Blunsom, A convolutional neural network for modelling sentences, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Baltimore, Maryland, 2014, pp. 655–665. `doi:10.3115/v1/P14-1062`.
URL `https://www.aclweb.org/anthology/P14-1062`

[4] Y. Zhang, B. C. Wallace, A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification, CoRR abs/1510.03820 (2015). `arXiv:1510.03820`.
URL `http://arxiv.org/abs/1510.03820`

[5] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of the 2014 Conference on Empirical Methods in Natural

Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1746–1751. doi:10.3115/v1/D14-1181. URL https://www.aclweb.org/anthology/D14-1181

[6] D. Kotzias, M. Denil, P. Blunsom, N. de Freitas, Deep multi-instance transfer learning, CoRR abs/1411.3128 (2014). arXiv:1411.3128.

[7] A. Graves, Generating sequences with recurrent neural networks, CoRR abs/1308.0850 (2013). arXiv:1308.0850.

[8] M. Schuster, K. K. Paliwal, A. General, Bidirectional recurrent neural networks, IEEE Transactions on Signal Processing (1997).

[9] T. Mikolov, S. Kombrink, L. Burget, J. Černocký, S. Khudanpur, Extensions of recurrent neural network language model, in: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2011, pp. 5528–5531.

[10] W. Zaremba, I. Sutskever, O. Vinyals, Recurrent Neural Network Regularization, arXiv e-prints (2014) arXiv:1409.2329arXiv:1409.2329.

[11] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, in: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Seattle, Washington, USA, 2013, pp. 1631–1642. URL https://www.aclweb.org/anthology/D13-1170

[12] K. S. Tai, R. Socher, C. D. Manning, Improved semantic representations from tree-structured long short-term memory networks, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Association for Computational Linguistics, Beijing, China, 2015, pp. 1556–1566. doi:10.3115/v1/P15-1150. URL https://www.aclweb.org/anthology/P15-1150

[13] R. Dewey, Introduction to Psychology, Wadsworth Publishing Company, 2004. URL https://books.google.com.tr/books?id=1MgbkgEACAAJ

[14] P. Agathangelou, I. Katakis, A hybrid deep learning network for modelling opinionated content, in: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC '19, ACM, New York, NY, USA, 2019, pp. 1051–1053.

[15] D. Cer, Y. Yang, S. yi Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, Y.-H. Sung, B. Strope, R. Kurzweil, Universal sentence encoder (2018). arXiv:1803.11175.

[16] C. dos Santos, M. Gatti, Deep convolutional neural networks for sentiment analysis of short texts, in: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, Dublin City University and Association for Computational Linguistics, Dublin, Ireland, 2014, pp. 69–78.
URL https://www.aclweb.org/anthology/C14-1008

[17] R. Collobert, J. Weston, A unified architecture for natural language processing: Deep neural networks with multitask learning, in: Proceedings of the 25th International Conference on Machine Learning, ICML '08, ACM, New York, NY, USA, 2008, pp. 160–167.

[18] R. Socher, C. C.-Y. Lin, A. Y. Ng, C. D. Manning, Parsing natural scenes and natural language with recursive neural networks, in: Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML'11, Omnipress, Madison, WI, USA, 2011, p. 129–136.

[19] R. Socher, B. Huval, C. D. Manning, A. Y. Ng, Semantic compositionality through recursive matrix-vector spaces, in: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Association for Computational Linguistics, Jeju Island, Korea, 2012, pp. 1201–1211.
URL https://www.aclweb.org/anthology/D12-1110

[20] O. Irsoy, C. Cardie, Deep recursive neural networks for compositionality in language, in: Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems, Vol. 27, Curran Associates, Inc., 2014, pp. 2096–2104.
URL https://proceedings.neurips.cc/paper/2014/file/2cfd4560539f887a5e420412b370b361-Paper.pdf

[21] M. Denil, A. Demiraj, N. Kalchbrenner, P. Blunsom, N. de Freitas, Modelling, Visualising and Summarising Documents with a Single Convolutional Neural Network, arXiv e-prints (2014) arXiv:1406.3830arXiv: 1406.3830.

[22] J. Deriu, M. Gonzenbach, F. Uzdilli, A. Lucchi, V. De Luca, M. Jaggi, SwissCheese at SemEval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision, in: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), Association for Computational Linguistics, San Diego, California, 2016, pp. 1124–1128. doi:10.18653/v1/S16-1173.
URL https://www.aclweb.org/anthology/S16-1173

[23] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, CoRR abs/1301.3781 (2013). arXiv: 1301.3781.

[24] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543.

[25] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, A. Joulin, Advances in pre-training distributed word representations, in: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018), 2018.

[26] D. Shen, G. Wang, W. Wang, M. R. Min, Q. Su, Y. Zhang, C. Li, R. Henao, L. Carin, Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 440–450. doi: 10.18653/v1/P18-1041.
URL https://www.aclweb.org/anthology/P18-1041

[27] P. Zhu, H. Jiang, C. Zhang, M. Liao, H. Hu, Improving convolutional network using k-max mechanism for sentiment analysis tasks, in: 2020 IEEE International Conference on Information Technology,Big Data and Artificial Intelligence (ICIBA), Vol. 1, 2020, pp. 772–779. doi:10.1109/ ICIBA50161.2020.9276869.

[28] M. E. Basiri, S. Nemati, M. Abdar, E. Cambria, U. R. Acharya, Abcdm: An attention-based bidirectional cnn-rnn deep model for sentiment analysis, Future Generation Computer Systems 115 (2021) 279–294. doi:https://doi.org/10.1016/j.future.2020.08.005.
URL https://www.sciencedirect.com/science/article/pii/S0167739X20309195

[29] T. Chen, R. Xu, Y. He, X. Wang, Improving sentiment analysis via sentence type classification using bilstm-crf and cnn, Expert Systems with Applications 72 (2017) 221–230. doi:https://doi.org/10.1016/j.eswa.2016.10.065.
URL https://www.sciencedirect.com/science/article/pii/S0957417416305929

[30] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.

[31] J. L. Elman, Finding structure in time, Cognitive Science 14 (2) (1990) 179–211.

[32] R. Pascanu, T. Mikolov, Y. Bengio, Understanding the exploding gradient problem, CoRR abs/1211.5063 (2012). arXiv:1211.5063.

[33] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Transactions on Neural Networks 5 (2) (1994) 157–166.

[34] M. Nabil, A. Atyia, M. Aly, CUFE at SemEval-2016 task 4: A gated recurrent model for sentiment classification, in: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), Association for Computational Linguistics, San Diego, California, 2016, pp. 52–57. doi:10.18653/v1/S16-1005.
URL https://www.aclweb.org/anthology/S16-1005

[35] A. Graves, N. Jaitly, A. r. Mohamed, Hybrid speech recognition with deep bidirectional lstm, in: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, 2013, pp. 273–278.

[36] B. Shu, F. Ren, Y. Bao, Investigating lstm with k-max pooling for text classification, in: 2018 11th International Conference on Intelligent

Computation Technology and Automation (ICICTA), 2018, pp. 31–34. doi:10.1109/ICICTA.2018.00015.

[37] C. Zhou, C. Sun, Z. Liu, F. C. M. Lau, A C-LSTM neural network for text classification, CoRR abs/1511.08630 (2015). arXiv:1511.08630. URL http://arxiv.org/abs/1511.08630

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, CoRR abs/1706.03762 (2017). arXiv:1706.03762. URL http://arxiv.org/abs/1706.03762

[39] Z. Lei, Y. Yang, M. Yang, Y. Liu, A multi-sentiment-resource enhanced attention network for sentiment classification, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 758–763. doi:10.18653/v1/P18-2120. URL https://www.aclweb.org/anthology/P18-2120

[40] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 2227–2237. doi:10.18653/v1/N18-1202. URL https://www.aclweb.org/anthology/N18-1202

[41] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners (2019).

[42] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423

[43] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, SemEval-2014 task 4: Aspect based sentiment analysis, in: Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), Association for Computational Linguistics, Dublin, Ireland, 2014, pp. 27–35. doi:10.3115/v1/S14-2004.
URL https://www.aclweb.org/anthology/S14-2004

[44] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, I. Androutsopoulos, SemEval-2015 task 12: Aspect based sentiment analysis, in: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Association for Computational Linguistics, Denver, Colorado, 2015, pp. 486–495. doi:10.18653/v1/S15-2082.
URL https://www.aclweb.org/anthology/S15-2082

[45] M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. AL-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq, V. Hoste, M. Apidianaki, X. Tannier, N. Loukachevitch, E. Kotelnikov, N. Bel, S. M. Jiménez-Zafra, G. Eryiğit, SemEval-2016 task 5: Aspect based sentiment analysis, in: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), Association for Computational Linguistics, San Diego, California, 2016, pp. 19–30. doi:10.18653/v1/S16-1002.
URL https://www.aclweb.org/anthology/S16-1002

[46] M. Denil, A. Demiraj, N. Kalchbrenner, P. Blunsom, N. de Freitas, Modelling, visualising and summarising documents with a single convolutional neural network, CoRR abs/1406.3830 (2014). arXiv:1406.3830.
URL http://arxiv.org/abs/1406.3830

[47] P. Agathangelou, I. Katakis, F. Kokkoras, K. Ntonas, Mining domain-specific dictionaries of opinion words, in: B. Benatallah, A. Bestavros, Y. Manolopoulos, A. Vakali, Y. Zhang (Eds.), Web Information Systems Engineering – WISE 2014, Springer International Publishing, Cham, 2014, pp. 47–62.

[48] P. Agathangelou, I. Katakis, I. Koutoulakis, F. Kokkoras, D. Gunopulos, Learning patterns for discovering domain-oriented opinion words, Knowledge and Information Systems (Jun 2017).

[49] I. Sutskever, J. Martens, G. Hinton, Generating text with recurrent neural networks, in: Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML'11, Omnipress, USA, 2011, pp. 1017–1024.

[50] A. Graves, N. Jaitly, A. Mohamed, Hybrid speech recognition with deep bidirectional LSTM, in: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013, IEEE, 2013, pp. 273–278.

[51] J. S. Bridle, Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition, in: Fogelman-Soulie, Herault (Eds.), Neurocomputing: Algorithms, Architectures and Applications, NATO ASI Series, Springer, 1990, pp. 227–236.

[52] D. Masters, C. Luschi, Revisiting small batch training for deep neural networks, CoRR abs/1804.07612 (2018). `arXiv:1804.07612`.

[53] A. Krogh, J. A. Hertz, A simple weight decay can improve generalization, in: Proceedings of the 4th International Conference on Neural Information Processing Systems, NIPS'91, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1991, pp. 950–957.

[54] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization (2014). `arXiv:1412.6980`.

[55] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, CoRR abs/1207.0580 (2012). `arXiv:1207.0580`.

[56] M. Munikar, S. Shakya, A. Shrestha, Fine-grained sentiment classification using bert, in: 2019 Artificial Intelligence for Transforming Business and Society (AITB), Vol. 1, 2019, pp. 1–5. `doi:10.1109/AITB48515.2019.8947435`.

[57] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, Q. V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing

Systems, Vol. 32, Curran Associates, Inc., 2019.
URL https://proceedings.neurips.cc/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf

[58] S. Wang, H. Fang, M. Khabsa, H. Mao, H. Ma, Entailment as Few-Shot Learner, arXiv e-prints (2021) arXiv:2104.14690arXiv:2104.14690.

[59] H. Jiang, P. He, W. Chen, X. Liu, J. Gao, T. Zhao, SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, 2020, pp. 2177–2190. doi:10.18653/v1/2020.acl-main.197.
URL https://www.aclweb.org/anthology/2020.acl-main.197

[60] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, A. Bordes, Supervised learning of universal sentence representations from natural language inference data, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Copenhagen, Denmark, 2017, pp. 670–680. doi:10.18653/v1/D17-1070.
URL https://www.aclweb.org/anthology/D17-1070

[61] M. Iyyer, V. Manjunatha, J. Boyd-Graber, H. Daumé III, Deep unordered composition rivals syntactic methods for text classification, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Association for Computational Linguistics, Beijing, China, 2015, pp. 1681–1691. doi:10.3115/v1/P15-1162.
URL https://www.aclweb.org/anthology/P15-1162

[62] Y. Wang, A. Sun, J. Han, Y. Liu, X. Zhu, Sentiment analysis by capsules, in: Proceedings of the 2018 World Wide Web Conference, WWW '18, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 2018, p. 1165–1174. doi:10.1145/3178876.3186015.
URL https://doi.org/10.1145/3178876.3186015

[63] H. Zhao, Z. Lu, P. Poupart, Self-adaptive hierarchical sentence model, in: Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15, AAAI Press, 2015, p. 4069–4076.

[64] S. Wang, C. Manning, Fast dropout training, in: S. Dasgupta, D. McAllester (Eds.), Proceedings of the 30th International Conference on Machine Learning, Vol. 28 of Proceedings of Machine Learning Research, PMLR, Atlanta, Georgia, USA, 2013, pp. 118–126.
URL http://proceedings.mlr.press/v28/wang13a.html

[65] F. Hill, K. Cho, A. Korhonen, Learning distributed representations of sentences from unlabelled data, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, San Diego, California, 2016, pp. 1367–1377. doi:10.18653/v1/N16-1162.
URL https://www.aclweb.org/anthology/N16-1162

[66] J. Mu, P. Viswanath, All-but-the-top: Simple and effective postprocessing for word representations, in: International Conference on Learning Representations, 2018.
URL https://openreview.net/forum?id=HkuGJ3kCb

[67] W. Zhao, J. Ye, M. Yang, Z. Lei, S. Zhang, Z. Zhao, Investigating capsule networks with dynamic routing for text classification, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 3110–3119. doi:10.18653/v1/D18-1350.
URL https://www.aclweb.org/anthology/D18-1350

[68] P. Xu, A. Madotto, C.-S. Wu, J. H. Park, P. Fung, Emo2Vec: Learning generalized emotion representation by multi-task training, in: Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 292–298. doi:10.18653/v1/W18-6243.
URL https://www.aclweb.org/anthology/W18-6243

[69] F. Wang, Z. Zhang, M. Lan, Ecnu at semeval-2016 task 7: An enhanced

supervised learning method for lexicon sentiment intensity ranking, in: SemEval@NAACL-HLT, 2016.

[70] S. Xu, H. Liang, T. Baldwin, UNIMELB at SemEval-2016 tasks 4A and 4B: An ensemble of neural networks and a Word2Vec based model for sentiment classification, in: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), Association for Computational Linguistics, San Diego, California, 2016, pp. 183–189. `doi:10.18653/v1/S16-1027`.
URL `https://www.aclweb.org/anthology/S16-1027`

[71] V. Yadav, thecerealkiller at SemEval-2016 task 4: Deep learning based system for classifying sentiment of tweets on two point scale, in: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), Association for Computational Linguistics, San Diego, California, 2016, pp. 100–102. `doi:10.18653/v1/S16-1013`.
URL `https://www.aclweb.org/anthology/S16-1013`

[72] G. Balikas, M.-R. Amini, TwiSE at SemEval-2016 task 4: Twitter sentiment classification, in: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), Association for Computational Linguistics, San Diego, California, 2016, pp. 85–91. `doi:10.18653/v1/S16-1010`.
URL `https://www.aclweb.org/anthology/S16-1010`

[73] D. Stojanovski, G. Strezoski, G. Madjarov, I. Dimitrovski, Finki at SemEval-2016 task 4: Deep learning architecture for Twitter sentiment analysis, in: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), Association for Computational Linguistics, San Diego, California, 2016, pp. 149–154. `doi:10.18653/v1/S16-1022`.
URL `https://www.aclweb.org/anthology/S16-1022`

[74] X. Li, D. Roth, Learning question classifiers, in: Proceedings of the 19th International Conference on Computational Linguistics - Volume 1, COLING '02, Association for Computational Linguistics, Stroudsburg, PA, USA, 2002, pp. 1–7.

[75] J. Otterbacher, I. Katakis, P. Agathangelou, Linguistic Bias in Crowdsourced Biographies: A Cross-lingual Examination, 2019, Ch. Chap-

ter 12, pp. 411–440. arXiv:https://www.worldscientific.com/doi/pdf/10.1142/9789813274884_0012.

[76] M. Dragoni, G. Petrucci, A neural word embeddings approach for multi-domain sentiment analysis, IEEE Transactions on Affective Computing 8 (4) (2017) 457–470. doi:10.1109/TAFFC.2017.2717879.

[77] M. Dragoni, A. Tettamanzi, C. da Costa Pereira, DRANZIERA: An evaluation protocol for multi-domain opinion mining, in: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), European Language Resources Association (ELRA), Portorož, Slovenia, 2016, pp. 267–272.
URL https://www.aclweb.org/anthology/L16-1041