A Helping Hand: Transfer Learning for Deep Sentiment Analysis

Xin Dong
Rutgers University
New Brunswick, NJ, USA
xd48@rutgers.edu

Gerard de Melo
Rutgers University
New Brunswick, NJ, USA
gdm@demelo.org

Abstract

Deep convolutional neural networks excel at sentiment polarity classification, but tend to require substantial amounts of training data, which moreover differs quite significantly between domains. In this work, we present an approach to feed generic cues into the training process of such networks, leading to better generalization abilities given limited training data. We propose to induce sentiment embeddings via supervision on extrinsic data, which are then fed into the model via a dedicated memory-based component. We observe significant gains in effectiveness on a range of different datasets in seven different languages.

1 Introduction

Over the past decades, sentiment analysis has grown from an academic endeavour to an essential analytics tool. Across the globe, people are voicing their opinion in online social media, product review sites, booking platforms, blogs, etc. Hence, it is important to keep abreast of ongoing developments in all pertinent markets, accounting for different domains as well as different languages. In recent years, deep neural architectures based on convolutional or recurrent layers have become established as the preeminent models for supervised sentiment polarity classification. At the same time, it is also frequently observed that deep neural networks tend to be particularly data-hungry. This is a problem in many real-world settings, where large amounts of training examples may be too costly to obtain for every target domain. A model trained on movie reviews, for instance, will fare very poorly on the task of assessing restaurant or hotel reviews, let alone tweets about politicians.

In this paper, we investigate how extrinsic signals can be incorporated into deep neural networks for sentiment analysis. Numerous papers have found the use of regular pre-trained word vector representations to be beneficial for sentiment analysis (Socher et al., 2013; Kim, 2014; dos Santos and de C. Gatti, 2014). In our paper, we instead consider word embeddings specifically specialized for the task of sentiment analysis, studying how they can lead to stronger and more consistent gains, despite the fact that the embeddings were obtained using out-of-domain data.

An intuitive solution would be to concatenate regular embeddings, which provide semantic relatedness cues, with sentiment polarity cues that are captured in additional dimensions. We instead propose a bespoke convolutional neural network architecture with a separate memory module dedicated to the sentiment embeddings. Our empirical study shows that the sentiment embeddings can lead to consistent gains across different datasets in a diverse set of domains and languages if a suitable neural network architecture is used.

2 Approach

2.1 Sentiment Embedding Computation

Our goal is to incorporate external cues into a deep neural network such that the network is able to generalize better even when training data is scarce. While in computer vision, weights pre-trained on ImageNet are often used for transfer learning, the most popular way to incorporate external information into deep neural networks for text is to draw on word embeddings trained on vast amounts of word context information (Mikolov et al., 2013; Pennington et al., 2014; Peters et al., 2018). Indeed, the semantic relatedness signals provided by such representations often lead to slightly improved results in polarity classification tasks (Socher et al., 2013; Kim, 2014; dos Santos and de C. Gatti, 2014).

However, the co-occurrence-based objectives of word2vec and GloVe do not consider sentiment

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 2524–2534
Melbourne, Australia, July 15 - 20, 2018.

specifically. We thus seek to examine how complementary sentiment-specific information from an external source can give rise to further gains.

Transfer Learning. To this end, our goal is to induce sentiment embeddings that capture sentiment polarity signals in multiple domains and hence may be useful across a range of different sentiment analysis tasks. The multi-domain nature of these distinguish them from the kinds of generic polarity scores captured in sentiment polarity lexicons. We achieve this via transfer learning from trained models, benefiting from supervision on a series of sentiment polarity tasks from different domains. Given a training collection consisting of n binary classification tasks (e.g., with documents in n different domains), we learn n corresponding polarity prediction models. From these, we then extract token-level scores that are tied to specific prediction outcomes. Specifically, we train n linear models $f_i(\mathbf{x}) = \mathbf{w}_i^\intercal \mathbf{x} + b_i$ for tasks $i = 1, \ldots, n$. Then, each vocabulary word index j is assigned a new n-dimensional word vector $\mathbf{x_j} = (w_{1,j}, \cdots, w_{n,j})$ that incorporates the linear coefficients for that word across the different linear models.

A minor challenge is that naïvely using bag-of-word features can lead to counter-intuitive weights. If a word such as "*pleased*" in one domain mainly occurs after the word "*not*", while the reviews in another domain primarily used "*pleased*" in its unnegated form, then "*pleased*" would be assessed as possessing opposite polarities in different domains. To avoid this, we assume that features are preprocessed to better reflect whether words occur in a negated context. In our experiments, we simply treat occurrences of "*not* ⟨word⟩" as a single feature "*not_*⟨word⟩". Of course, one can replace this heuristic with much more sophisticated techniques that fully account for the scope of a wider range of negation constructions.

Graph-Based Extension. Most sentiment-related resources are available for the English language. To produce vectors for other languages in our experiments, we rely on cross-lingual projection via graph-based propagation (de Melo, 2015; de Melo, 2017; Dong and de Melo, 2018). At this point, we have a set of initial sentiment embedding vectors $\tilde{\mathbf{v}}_x \in \mathbb{R}^n$ for words $x \in V_0$. We assume that we have a lexical knowledge graph $G_\mathrm{L} = (V, A_\mathrm{L})$ with a node set consisting of an extended multilingual vocabulary $V \supseteq V_0$ and a set of weighted directed arcs $A_\mathrm{L} = \{(x_1, x'_1, w_1), \ldots, (x_m, x'_m, w_m)\}$. Each such arc reflects a weighted semantic connection between two vocabulary items $x, x' \in V$, where vocabulary items are words labeled with their respective language. Typically, many of the arcs in the G_L would reflect translational equivalence, but in our experiments, we also include monolingual links between semantically related words. Given this data, we aim to minimize

$$-\sum_{x \in V} \mathbf{v}_x^\intercal \left[\frac{1}{\sum\limits_{(x,x',w) \in A_\mathrm{L}} w} \sum_{(x,x',w) \in A_\mathrm{L}} w \mathbf{v}_{x'} \right] + C \sum_{x \in V_0} \|\mathbf{v}_x - \tilde{\mathbf{v}}_x\|_2 \quad (1)$$

The first component of this objective seeks to ensure that sentiment embeddings of words accord with those of their connected words, in terms of the dot product. The second part ensures that the deviation from any available initial word vectors $\tilde{\mathbf{v}}_x$ is minimal (for some very high constant C). For optimization, we preinitialize $\mathbf{v}_x = \tilde{\mathbf{v}}_x$ for all $x \in V_0$, and then rely on stochastic gradient descent steps.

2.2 Dual-Module Memory based CNNs

To feed this sentiment information into our architecture, we propose a Dual-Module Memory based Convolutional Neural Network (DM-MCNN) approach, which incorporates a dedicated memory module to process the sentiment embeddings, as illustrated in Fig. 1. While the module with regular word embeddings enables the model to learn salient patterns and harness the nearest neighbour and linear substructure properties of word embeddings, we conjecture that a separate sentiment memory module allows for better exploiting the information brought to the table by the sentiment embeddings.

Convolutional Module Inputs and Filters. The Convolutional Module input of the DM-MCNN is a sentence matrix $\mathbf{S} \in \mathbb{R}^{s \times d}$, the rows of which represent the words of the input sentence after tokenization. In the case of $\mathbf{S}$, i.e., in the regular module, each word is represented by its conventional word vector representation. Here, s refers to the length of a sentence, and d represents the dimensionality of the regular word vectors.

We perform convolutional operations on these matrices via linear filters. Given rows representing discrete words, we rely on weight matrices $\mathbf{W} \in \mathbb{R}^{h \times d}$ with region size h. We use the notation $\mathbf{S_{i:j}}$ to denote the sub-matrix of $\mathbf{S}$ from row i to row

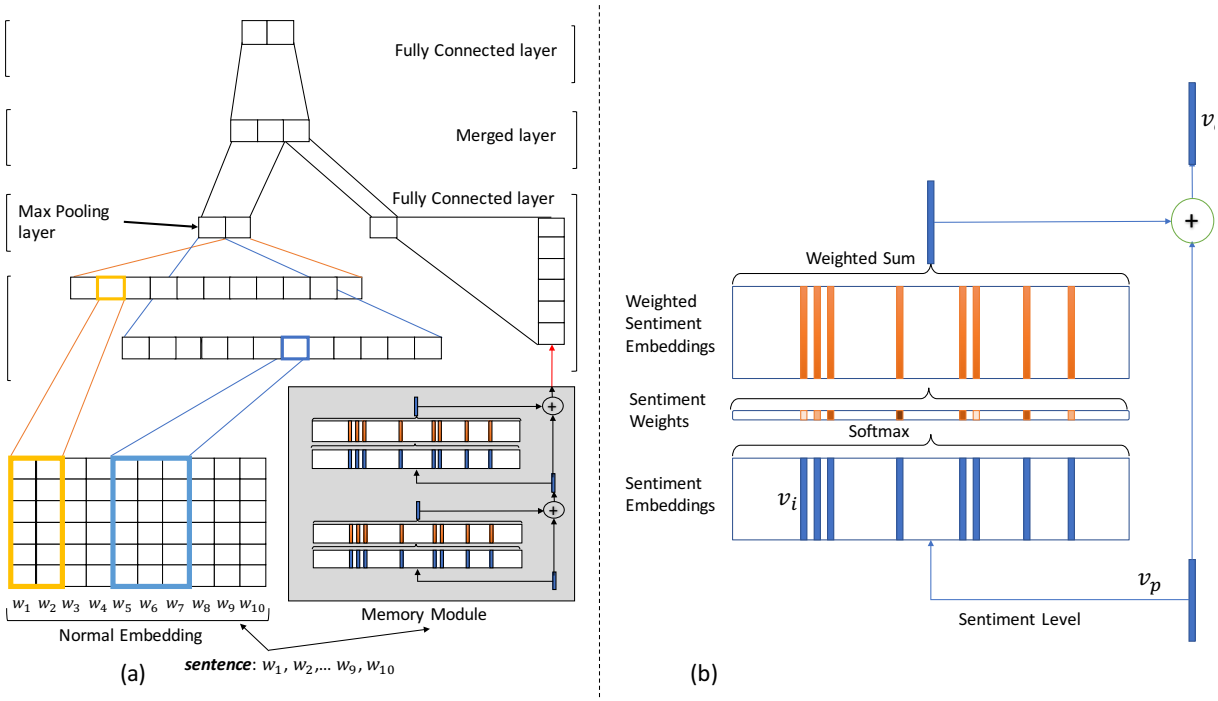

Figure 1: (a) Dual-Module Memory based Convolutional Neural Network architecture. (b) Single layer in Memory Module

j. Supposing that the weight matrix has a filter width of h, a wide convolution (Kalchbrenner et al., 2014) is induced such that out-of-range submatrix values $S_{i,j}$ with $i < 1$ or $i > s$ are taken to be zero. Thus, applying the filter on sub-matrices of $\mathbf{S}$ yields the output sequence $\mathbf{o} \in \mathbb{R}^{s+h-1}$ as

$$o_i = \mathbf{W} \odot \mathbf{S_{i:i+h-1}}, \tag{2}$$

where the $\odot$ operator provides the sum of an element-wise multiplication. Wide convolutions ensure that filters can cover words at the margins of the normal weight matrix.

Next, the c_i in feature maps $\mathbf{c} \in \mathbb{R}^{s+h-1}$ are computed as: $c_i = f(o_i + b)$, where $i = 1, \ldots, s + h - 1$, the parameter $b \in \mathbb{R}$ is a bias term, and f is an activation function.

Multiple Layers in Memory Module. The memory module obtains as input a sequence of sentiment embedding vectors for the input, and attempts to draw conclusions about the overall sentiment polarity of the entire input sequence. Given a set of sentence words $S = \{w_1, w_2, w_3, \ldots, w_n\}$, each word is mapped to its sentiment embedding vector of dimension d_s and we denote this set of vectors as V_s. The preliminary sentiment level $\mathbf{v}_\mathrm{p}$ is also a vector of dimensionality d_s. We take the mean of all sentiment vectors $\mathbf{v}_i$ for words $w_i \in S$ to initialize $\mathbf{v}_\mathrm{p}$. Next, we compute a vector $\mathbf{s}$ of similarities s_i between $\mathbf{v}_\mathrm{p}$ and each sentiment word vector $\mathbf{v}_i$, by taking the inner product, followed by ℓ_2-normalization and a softmax:

$$s_i = \frac{\exp \frac{\mathbf{v}_\mathrm{p}^\intercal \mathbf{v}_i}{\|\mathbf{v}_\mathrm{p}^\intercal \mathbf{v}_i\|_2}}{\sum_i \exp \frac{\mathbf{v}_\mathrm{p}^\intercal \mathbf{v}_i}{\|\mathbf{v}_\mathrm{p}^\intercal \mathbf{v}_i\|_2}} \tag{3}$$

As the sentiment embeddings used in our paper are generated from a linear model, the degree of correspondence between $\mathbf{v}_\mathrm{p}$ and $\mathbf{v}_i$ can adequately be assessed by the inner product. The resulting vector of scores $\mathbf{s}$ can be regarded as yielding sentiment weights for each word in the sentence. We apply ℓ_2-normalization to ensure a more balanced weight distribution. The output sentiment level vector $\mathbf{v}_\mathrm{o}$ is then a sum over the sentiment inputs $\mathbf{v}_i$ weighted by the ℓ_2-normalized vector of similarities:

$$\mathbf{v}_\mathrm{o} = \sum_i \frac{s_i}{\|\mathbf{s}\|_2} \mathbf{v}_i \tag{4}$$

This processing can be repeated in multiple passes, akin to how end-to-end memory networks for question answering often perform multiple hops (Sukhbaatar et al., 2015). While in the first iteration, $\mathbf{v}_\mathrm{p}$ was set to the mean sentiment vector, subsequent passes may allow us to iteratively refine

this vector. Assuming that $\mathbf{v}_{\mathrm{o}}^{k}$ has been produced by the k-th pass, then the subsequent level $\mathbf{v}_{\mathrm{p}}^{k+1}$ in the next pass is:

$$\mathbf{v}_{\mathrm{p}}^{k+1} = \mathbf{v}_{\mathrm{o}}^{k} + \mathbf{v}_{\mathrm{p}}^{k} \tag{5}$$

The intuition here is that multiple passes can enable the model to adaptively retrieve iterative sentiment level statistics beyond the initial average sentiment information.

Merging Layer and Prediction. Subsequently, for the convolutional module, 1d-max pooling is applied to $\mathbf{c}$, which ought to capture the most prominent signals. In the memory module, the final sentiment vector is modulated by a weight matrix $\mathbf{W}_{\mathrm{s}} \in \mathbb{R}^{l \times d_{\mathrm{s}}}$ to form a feature vector of dimensionality l. In general, we can use multiple filters to obtain several features in the convolutional module, while the memory module allows for adjusting the number of passes over the memory.

Finally, the outputs of these two modules are concatenated to form a fixed-length vector, which is passed to a fully connected softmax layer to obtain the final output probabilities.

Loss Function and Training. Our loss function is the cross-entropy function

$$\mathcal{L} = -\frac{1}{n} \sum_{i=1}^{n} \sum_{c \in C} y_{i,c} \ln \hat{y}_{i,c}, \tag{6}$$

where n is the number of training examples, C is the set of (two) classes, $y_{i,c}$ are ground truth labels for a given training example and class c, and $\hat{y}_{i,c}$ are corresponding label probabilities predicted by the model, as emitted by the softmax layer. We train our model using Adam optimization (Kingma and Ba, 2014) for better robustness across different datasets. Further details about our training regime follow in the Experiments section.

3 Experiments

We now turn to our extensive empirical evaluation, which assesses the effectiveness of our novel architecture with sentiment word vectors.

3.1 Experimental Setup

Datasets. For evaluation, we use real world datasets for 7 different languages, taken from a range of different sources that cover several domains. These are summarized in Table 1, with ISO 639-3 language codes. In our experimental setup, these are all cast as binary polarity classification tasks, for which we use accuracy as our evaluation metric.

Table 1: Dataset Descriptions

Language	Source	Domain	train	test
en	SST	Movies	6,920	1,821
	AFF	Food	5,945	1,189
es	SE16-T5	Restaurants	2,070	881
ru	TA	Hotels	2,387	682
de	TA	Restaurants	1,687	481
cs	TA	Restaurants	1,722	491
it	TA	Hotels	3,437	982
ja	TA	Restaurants	1,435	411

- The Stanford Sentiment Treebank (**SST**) dataset (Socher et al., 2013) consists of movie reviews taken from the Rotten Tomatoes website, including binary labels. We only used sentence-level data in our experiment.
- The SemEval-2016 Task 5 (**SE16-T5**) dataset (Pontiki et al., 2016) provides Spanish reviews of restaurants. It targeted aspect-based sentiment analysis, so we converted the entity-level annotations to sentence-level polarity labels via voting. As the number of entities per sentence is often one or very low, this process is reasonably precise. In any case, it enables us to compare the ability of different model variants to learn to recognize pertinent words.
- From TripAdvisor (**TA**), we crawled German, Russian, Italian, Czech, and Japanese reviews of restaurants and hotels. We removed three-star reviews, as these can be regarded as neutral ones, so reviews with a rating < 3 are considered negative, while those with a rating > 3 were deemed positive.
- The Amazon Fine Food Reviews **AFF** (McAuley and Leskovec, 2013) dataset provides food reviews left on Amazon. We chose a random subset of it with preprocessing as for TripAdvisor.

As there was no test set provided for TripAdvisor or for the Amazon Fine Food Reviews data, we randomly partitioned this data into training, validation, and test splits with a 80%/10%/20% ratio. Additionally, 10% of the training sets from SE16-T5 were randomly extracted and reserved for validation, while SST provides its own validation set. The new datasets are available from `http://gerard.demelo.org/sentiment/`.

Embeddings. The standard pre-trained word vectors used for English are the GloVe (Pennington et al., 2014) ones trained on 840 billion tokens of

Table 2: DM-MCNN Model Parameter Settings.

(a) General configuration.

Description		Values
Convol. Module	filter region size	(3,4,5)
	feature maps	100
	pooling	1d-max pooling
Memory Module	# passes (k)	2
	feature vector size	100
dropout rate		0.5
optimizer		Adam
activation function		ReLU
batch size		50

(b) Learning rate α used in DM-MCNN under 7 languages.

	en	es	de	ru
α	0.0004	0.0008	0.003	0.003
	cs	it	ja	
α	0.003	0.003	0.003	

Common Crawl data[1], while for other languages, we rely on the Facebook fastText Wikipedia embeddings (Bojanowski et al., 2016) as input representations. All of these are 300-dimensional. The vectors are either fed to the CNN, or to the convolutional module of the DM-MCNN during initialization, while unknown words are initialized with zeros. All words, including the unknown ones, are fine-tuned during the training process.

For our transfer learning approach, our experiments rely on the multi-domain sentiment dataset by Blitzer et al. (2007), collected from Amazon customers reviews. This dataset includes 25 categories of products and is used to generate our sentiment embeddings using linear models. Specifically, we train linear SVMs using scikit-learn to extract word coefficients in each domain and also for the union of all domains together, yielding a 26-dimensional sentiment embedding.

For comparison and analysis, we also consider several alternative forms of infusing external cues. Firstly, lexicon-driven methods have often been used for domain-independent sentiment analysis. We consider a recent sentiment lexicon called VADER (Hutto and Gilbert, 2014). The polarity scores assigned to words by the lexicon are taken as the components of a set of 1-dimensional word vectors (dividing the original scores by the difference between max and min polarity scores for normalization). Secondly, as another particularly strong alternative, we consider the SocialSent Reddit community-specific lexicons mined by the Stanford NLP group (Hamilton et al., 2016). These contain separate domain-specific scores for 250 different Reddit communities, and hence result in 250-dimensional embeddings.

For cross-lingual projection, we extract links between words from a 2017 dump of the English edition of Wiktionary. We restrict the vocabulary link set to include the languages in Table 1, mining corresponding translation, synonymy, derivation, and etymological links from Wiktionary.

Neural Network Details. For CNNs, we make use of the well-known CNN-non-static architecture and hyperparameters proposed by Kim (2014), with a learning rate of 0.0006, obtained by tuning on the validation data. For our DM-MCNN models, the configuration of the convolutional module is the same as for CNNs, and the remaining hyperparameter values were as well tuned on the validation sets. An overview of the relevant network parameter values is given in Table 2.

For greater efficiency and better convergence properties, the training relies on mini-batches. Our implementation considers the maximal sentence length in each mini-batch and zero-pads all other sentences to this length under convolutional module, thus enabling uniform and fast processing of each mini-batch. All neural network architectures are implemented using the PyTorch framework[2].

3.2 Results and Analysis

Baseline Results. Our main results are summarized in Table 3. We compare both regular CNNs and our dual-module alternative DM-MCNNs under a variety of settings. A common approach is to use a CNN with randomly initialized word vectors. Comparing this to CNNs with GloVe/fastText embeddings, where GloVe is used for English, and fastText is used for all other languages, we observe substantial improvements across all datasets. This shows that word vectors do tend to convey pertinent word semantics signals that enable models to generalize better. Note also that the accuracy using GloVe on the English movies review dataset is consistent with numbers reported in previous work (Zhang and Wallace, 2015).

Dual-Module Architecture. Next, we consider our DM-MCNNs with their dual-module mechanism to take advantage of transfer learning. We observe fairly consistent and sometimes quite substan-

[1] https://nlp.stanford.edu/projects/glove/

[2] http://pytorch.org

Table 3: Accuracy on several different English and non-English datasets from different domains, comparing our architecture against CNNs. Rest.: restaurants domain.

Approach	d	en		es	ru	de	cs	it	ja
		Movies	Food	Rest.	Hotels	Rest.	Rest.	Hotels	Rest.
CNN									
— Random Init.	300	80.78	86.63	81.50	90.18	88.09	90.00	93.18	78.59
— Word Vec. Init.	300	85.72	87.97	85.13	92.82	92.10	92.46	96.20	77.62
Our Approach									
— With fine-tuning	300/26	**86.99**	**90.08**	85.02	93.40	93.14	93.08	95.50	**85.40**
— No fine-tuning	300/26	86.38	88.81	**85.70**	**94.87**	**94.59**	**93.48**	96.20	77.62
CNN with Concatenated Sentiment Embeddings									
— VADER	301	85.89	88.39	84.90	92.31	88.36	93.08	96.34	77.62
— SocialSent	550	84.90	88.48	82.63	92.23	91.48	86.56	94.51	76.64
— Our Embeddings	326	86.05	89.07	84.56	92.72	93.56	91.24	95.78	77.62
Our Model with Alternative Sentiment Embeddings									
— Random	300/26	86.16	87.97	85.24	93.99	93.14	92.67	96.20	80.29
— VADER	300/1	86.33	88.39	84.45	94.18	92.31	92.87	96.48	75.43
— SocialSent	300/250	86.38	87.89	83.09	93.40	92.31	93.28	**96.62**	81.02

tial gains over CNNs with just the GloVe/fastText vectors. We see that the sentiment embeddings provide important complementary signals beyond what is provided in regular word embeddings, and that our dual-module approach succeeds at exploiting these signals across a range of different domains and languages. Our transfer learning approach leads to sentiment embeddings that capture signals from multiple domains. The model successfully picks the pertinent parts of this signal for datasets from domains as different as movie reviews and food reviews.

We report results for two different training conditions. In the first condition (with fine-tuning), the sentiment embedding matrix is preinitialized using the data from our transfer learning procedure, but the model is then able to modify these arbitrarily via backpropagation. In the second condition (no fine-tuning), we simply use our sentiment embedding matrix as is, and do not update it. Instead, the model is able to update its various other parameters, particularly its various weight matrices and bias vectors. While both training conditions outperform the CNN baseline, there is no obvious winner among the two. When the training data set is very small and hence there is a significant risk of overfitting, one may be best advised to forgo fine-tuning. In contrast, when it is somewhat larger (as for our English datasets, which each have over 5,000 training instances) or when the language is particularly idiosyncratic or not covered sufficiently well by our cross-lingual projection procedure (such as perhaps for Japanese), then fine-tuning is recommended. In this case, fine-tuning may allow the model to adjust the embeddings to cater to domain-specific meanings and corpus-specific correlations, while also overcoming possible sparsity of the cross-lingual vectors resulting from a lack of coverage of the translation dictionary.

It is important to note that many of the results in Table 3 stem from embeddings that were created automatically using cross-lingual projection. Our transfer learning embeddings were induced from entirely English data. Although the automatically projected cross-lingual embeddings are very noisy and limited in their coverage, particularly with respect to inflected forms, our model succeeds in exploiting them to obtain substantial gains in several different languages and domains.

Alternative Embedding Methods. For a more detailed analysis, we conducted additional experiments with alternative embedding conditions. In particular, as a simpler means of achieving gains over standard CNNs, we propose to use CNNs with word vectors augmented with sentiment cues. Given that regular word embeddings appear to be useful for capturing semantics, one may conjecture that extending these word vectors with additional dimensions to capture sentiment information can lead to improved results. For this, we simply concatenate the regular word embeddings with different forms of sentiment embeddings that we have obtained, including those from the sentiment lexicon VADER, from the Stanford SocialSent project, and from our transfer learning procedure via Amazon reviews. To conduct these experiments, we also produced cross-lingual projections of the VADER and SocialSent embedding data.

The results of using these embeddings as opposed to regular ones are somewhat mixed. Con-

catenating the VADER embeddings or our transfer learning ones leads to minor improvements on English, and our cross-lingual projection of them leads to occasional gains, but the results are far from consistent. Even on English, adding the 250-dimensional SocialSent embedding seems to degrade the effectiveness of the CNN, although all input information that was previously there continues to be provided to it. This suggests that a simple concatenation may harm the model's ability to harness the semantic information carried by regular word vectors. This risk seems more pronounced for larger-dimensional sentiment embeddings.

In contrast, with our DM-MCNNs approach, the sentiment information is provided to the model in a separate memory module that makes multiple passes over this data before combining it with the regular CNN module's signals. Thus, the model can exploit the two kinds of information independently, and learn a suitable way to aggregate them to produce an overall output classification.

This hence demonstrates not only that the sentiment embeddings tend to provide important complementary signals but also that a dual-module approach is best-suited to incorporate such signals into deep neural models.

We also analysed our DM-MCNNs with alternative embeddings. When we feed random sentiment embeddings into them, not unexpectedly, in many cases the results do not improve much. This is because our memory module has been designed to leverage informative prior information and to reweight its signals based on this assumption. Hence, it is important to feed genuine sentiment cues into the memory module. Yet, on some languages, we nevertheless note improvements over the CNN baseline. In these cases, even if similarities between pairs of sentiment vectors initially do not carry any significance, backpropagation may have succeeded in updating the sentiment embedding matrix such that eventually the memory module becomes able to discern salient patterns in the data.

We also considered our DM-MCNNs when feeding the VADER or SocialSent embeddings into the memory module. In this case, it also mostly succeeded in outperforming the CNN baseline. In fact, on the Italian TripAdvisor dataset, the SocialSent embeddings yielded the overall strongest results. In all other cases, however, our transfer learning embeddings proved more effective. We believe that this is because they are obtained in a data-driven manner based on an objective that directly seeks to optimize for classification accuracy.

Influence of Training Set Size. To look into the effect of our approach with restricted training data, we first consider the SST dataset as an instructive example. We set the training set size to 20%, 50%, 100% of its original size and compared our full dual module model with sentiment embeddings against state-of-the-art methods.

The results are given in Table 4. Our dual module CNN has a sizeable lead over other methods when only using 20% of SST training set. Given that we study how to incorporate extrinsic cues into a deep neural model, we consider CNN-Rule-q (Hu et al., 2016) and Gumbel Tree-LSTM (Choi et al., 2017) as the relevant baseline methods. The CNN-Rule-q method used an iterative distillation method that exploits structured information from logical rules, which for SST is based on the word *but* to determine the weights in the neural network. The Gumbel Tree-LSTM approach incorporates a Straight-Through Gumbel-Softmax into a tree-structured LSTM architecture that learns how to compose task-specific tree structures starting from plain raw text. They all require a large amount of data to pick up sufficient information during training, while our method is able to efficiently capture sentiment information from our transfer learning even though the data is scarce.

For further analysis, we also artificially reduce the training set sizes to 50% of the original sizes given in Table 1 for our multilingual datasets. The results are plotted in Fig. 2. We compare: 1) the CNN model baseline, 2) the CNN model but concatenating our sentiment embeddings from transfer learning, and 3) our full dual module model with these sentiment embeddings. We already saw in Table 3 that we obtain reasonable gains over generic embeddings when using the full training sets.

In Fig. 2, we additionally observe that the gains are overall much more remarkable on smaller training sets. This shows that the sentiment embeddings are most useful when they are of high quality and domain-specific training data is scarce, although a modest amount of training data is still needed for the model to be able to adapt to the target domain.

Inspection of the DM-MCNN-learned Deep Sentiment Information. To further investigate what the model is learning, we examine the changes of weights of words on the English SST dataset when using the VADER sentiment embeddings

Table 4: Accuracy on SST with increasing training sizes

Model	*20%*	*50%*	*100%*
CNN (Kim, 2014)	83.14	84.29	85.72
CNN-Rule-q (Hu et al., 2016)	83.75	85.45	86.49
Gumbel Tree-LSTM (Choi et al., 2017)	84.04	84.83	86.80
DC-MCNN (ours)	**85.06**	**86.16**	**86.99**

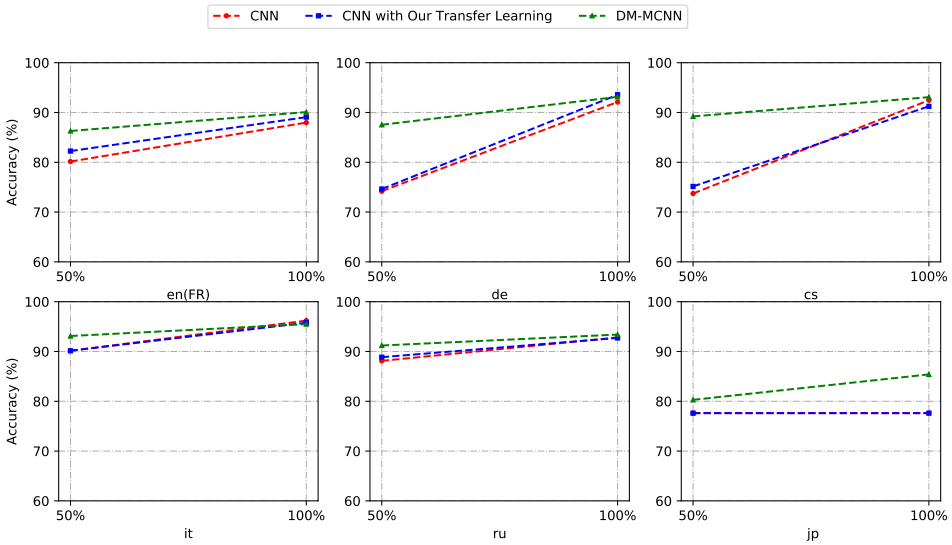

Figure 2: Effectiveness of three embedding alternatives on 6 languages at a reduced training size (comparing 50% and 100%).

with DM-MCNNs. Although these are not as powerful as our transfer learning embeddings, the VADER embeddings are the most easily interpretable here, since they are one-dimensional, and thus can be regarded as word-specific weights. The result is visualized in Fig. 3. Here, the dark-shaded segments (in blue) refer to the original weights, while the light-shaded segments (in red) refer to the adjusted weights after training. The medium-shaded segments (in purple) reflect the overlap between the two. Hence, whenever we observe a dark (blue) segment above a medium (purple) segment in a bar, we can infer that the fine-tuned weight for a word (e.g., for "*plays*" in Fig. 3) was lower than the original weight of that word. Conversely, whenever we observe a light (red) segment at the top, the weight increased during training (e.g., for *hilarious*). Generally, dark (blue) segments reflect decreased weight magnitudes and light (red) ones reflect increased weight magnitudes, both on the positive and on the negative side.

We consider in Fig. 3 the top 50 weight changes only of words that were already covered by the original VADER sentiment embeddings. Here, it is worth noting that the weight magnitudes of positive words such as "*laugh*", "*appealing*" and negative words such as "*lack*", "*missing*" increase further, while words such as "*damn*", "*interest*", "*war*" see decreases in magnitude, presumably due to their ambiguity and context (e.g., "*damn good*", "*lost the interest*", descriptions of war movies). Hence, the figure confirms that our DM-MCNN approach is able to exploit and customize the provided sentiment weights for the target domain. However, unlike the VADER data, our transfer learning approach results in multi-dimensional sentiment embeddings that can more easily capture multiple domains right from the start, thus making it possible to use them even without further fine-tuning.

4 Related Work

Sentiment Mining and Embeddings. There is a long history of work on collecting word polarity scores manually (Hu and Liu, 2004) or via graph-based propagation from seeds (Kim and Hovy, 2004; Baccianella et al., 2010). Maas et al. (2011) present a probabilistic topic model that exploits sentiment supervision during training, leading to rep-

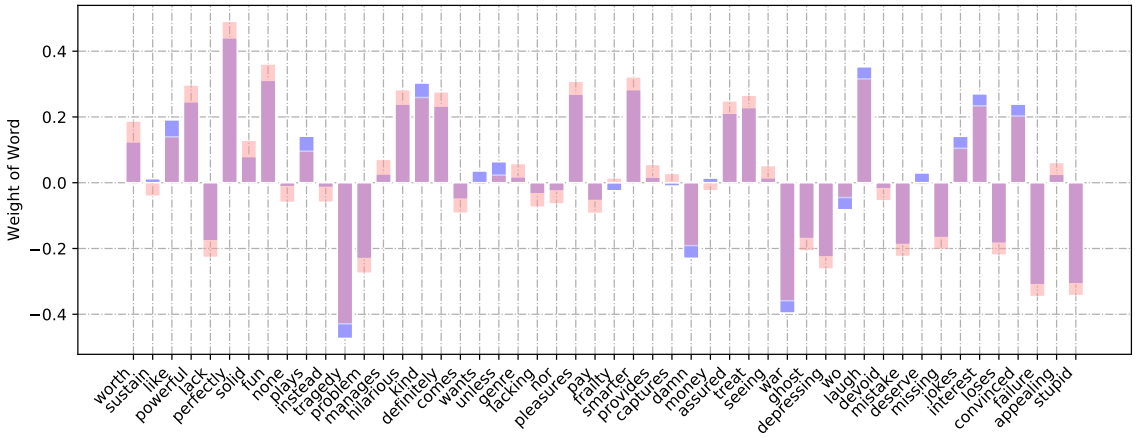

Figure 3: Top 50 weight changes of words fine-tuned by the sentiment memory module of the DM-MCNN, using the one-dimensional VADER embeddings, but considering only words with non-zero values in the original VADER data. Here, the dark shade (blue) refers to the original value of word vectors, while the light shade (red) refers to their fine-tuned values after training. The medium intensity (purple) corresponds to the overlap between the original and fine-tuned word vectors.

resentations that include sentiment signals. However, in their experiments, the semantic-only models mostly outperform the corresponding full models with extra sentiment signals. Tang et al. (2014) showed that one can acquire sentiment information by learning from millions of training examples via distant supervision. While prior work used such signals for rule-based sentiment analysis or for feature engineering in SVMs and other shallow models, our study examines how they are best be incorporated into deep neural models, as the baseline of naïvely feeding them into the model does not work sufficiently well.

Neural Architectures. In terms of architectures, deep recursive neural networks (Socher et al., 2013) were soon outperformed by deep convolutional and recurrent neural networks (İrsoy and Cardie, 2014; Kim, 2014). Recent work has investigated more involved models, with ingredients such as Tree-LSTMs (Tai et al., 2015; Looks et al., 2017), hierarchical attention (Yang et al., 2016), user and product attention (Chen et al., 2016), aspect-specific modeling (Wang et al., 2015), and part of speech-specific transition functions (Huang et al., 2017). Large ensemble models also tend to outperform individually trained sentiment analysis models (Looks et al., 2017). The goal of our study is not necessarily to devise the most sophisticated state-of-the-art neural architecture, but to demonstrate how external sentiment cues can be incorporated such architectures. Our initial explorations relied on a simple dual-channel convolutional neural network (Dong and de Melo, 2018). The present work proposes a more sophisticated approach, drawing on ideas from attention mechanisms in machine translation (Bahdanau et al., 2014) as well as from memory networks (Weston et al., 2014) and iterative attention (Kumar et al., 2015), which have proven useful for tasks such as question answering. We incorporate these ideas into a separate memory module that operates alongside the regular convolutional module.

5 Conclusions

Deep neural networks are widely used in sentiment polarity classification, but suffer from their dependence on very large annotated training corpora. In this paper, we study how to incorporate extrinsic cues into the network, beyond just generic word embeddings. We have found that this is best achieved using a dual-module approach that encourages the learning of models with favourable generalization abilities. Our experiments show that this can lead to gains across a number of different languages and domains. Our embeddings and multilingual datasets are freely available from `http://gerard.demelo.org/sentiment/`.

Acknowledgments

This research is funded in part by ARO grant no. W911NF-17-C-0098 as part of the DARPA SocialSim program.

References

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*. European Language Resources Association.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

John Blitzer, Mark Dredze, Fernando Pereira, et al. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, volume 7, pages 440–447.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. 2016. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1650–1659, Austin, Texas. Association for Computational Linguistics.

Jihun Choi, Kang Min Yoo, and Sang-goo Lee. 2017. Unsupervised learning of task-specific tree structures with tree-lstms. *arXiv preprint arXiv:1707.02786*.

Gerard de Melo. 2015. Wiktionary-based word embeddings. In *Proceedings of MT Summit XV*.

Xin Dong and Gerard de Melo. 2018. Cross-lingual propagation for deep sentiment analysis. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI 2018)*. AAAI Press.

William L. Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. 2016. Inducing domain-specific sentiment lexicons from unlabeled corpora. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 595–605, Austin, Texas. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *KDD 2004: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, New York, NY, USA. ACM.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. *arXiv preprint arXiv:1603.06318*.

Minlie Huang, Qiao Qian, and Xiaoyan Zhu. 2017. Encoding syntactic knowledge in neural networks for sentiment classification. *ACM Trans. Inf. Syst.*, 35(3):26:1–26:27.

C.J. Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proc. ICWSM-14*.

Ozan İrsoy and Claire Cardie. 2014. Opinion mining with deep recurrent neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 720–728.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

Soo-Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *Proceedings of Coling 2004*, pages 1367–1373, Geneva, Switzerland. COLING.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. 2015. Ask me anything: Dynamic memory networks for natural language processing. *CoRR*, abs/1506.07285.

Moshe Looks, Marcello Herreshoff, DeLesley Hutchins, and Peter Norvig. 2017. Deep learning with dynamic computation graphs. *CoRR*, abs/1702.02181.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 142–150. Association for Computational Linguistics.

Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*, pages 897–908. ACM.

Gerard de Melo. 2017. Inducing conceptual embedding spaces from Wikipedia. In *Proceedings of WWW 2017*. ACM.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. 2018. Deep contextualized word representations. *ArXiv e-prints*.

Maria Pontiki, Dimitrios Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, AL Mohammad, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. *Proceedings of SemEval*, pages 19–30.

Cícero Nogueira dos Santos and Maíra A. de C. Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING 2014*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *CoRR*, abs/1503.00075.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1555–1565, Baltimore, Maryland. Association for Computational Linguistics.

Linlin Wang, Kang Liu, Zhu Cao, Jun Zhao, and Gerard de Melo. 2015. Sentiment-aspect extraction based on Restricted Boltzmann Machines. In *Proceedings of ACL 2015*, pages 616–625.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory networks. *arXiv preprint arXiv:1410.3916*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Ye Zhang and Byron Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*.