Learning representations for sentiment classification using Multi-task framework

Hardik Meisheri
TCS Research
Mumbai, India
hardik.meisheri@tcs.com

Harshad Khadilkar
TCS Research
Mumbai, India
harshad.khadilkar@tcs.com

Abstract

Most of the existing state of the art sentiment classification techniques involve the use of pre-trained embeddings. This paper postulates a generalized representation that collates training on multiple datasets using a Multi-task learning framework. We incorporate publicly available, pre-trained embeddings with Bidirectional LSTM's to develop the multi-task model. We validate the representations on an independent test Irony dataset that can contain several sentiments within each sample, with an arbitrary distribution. Our experiments show a significant improvement in results as compared to the available baselines for individual datasets on which independent models are trained. Results also suggest superior performance of the representations generated over Irony dataset.

1 Introduction

Sentiment analysis has attracted substantial research interest, especially in the field of social media, owing to the growing number of data and active users. In addition, the research community has gravitated towards a pragmatic characterization of language with the division into (and quantification of) specific emotions for sentiment analysis. This approach has come to prominence in recent times as a large number of enterprises (not just social media corporations) now rely on understanding customer sentiments for defining product and marketing strategies (Pang and Lee, 2004; Socher et al., 2012).

Beyond strategic inputs, sentiment analysis also performs a tactical role in the age of rapid (viral) increases and decreases in the visibility of specific events, with magnified consequences for corporations and communities at large. For example, United Airlines faced significant business impact due to a single (possibly isolated) passenger-related incident, due to its spread over Twitter[1]. It is conceivable that an automated system quickly alerting the management about the rate and depth of negative sentiments due to the incident, would have enabled them to produce a more amelioratory response from the outset.

Complementary to such motivating incidents is the recent availability of large datasets from social media sources. Twitter has become a go-to choice for scraping data due to its large user base and the easy accessibility of tweets through its API. The result is a large corpus of complex sentiments for identification and analysis. Tweets (the messages posted on Twitter) are limited to 140 characters, which creates a plethora of challenges as the users find new and innovative ways of condensing the messages using slang, hashtags, and emojis, often defying traditional grammatical rules of the language. This is further complicated by the fast, localized rise and decay of popular memes, slang, and hashtags.

Traditional sentiment analysis using dictionary-based methods has failed to capture these nuances, as the methods rely on grammatically correct, intact syntactic and semantic structures which are not followed in this space. Traditional sentiment analyzers such as (Akkaya et al., 2009; Poria et al., 2014; Sharma and Bhattacharyya, 2013) that worked well with well-written texts, face challenges at lexical, syntactic and semantic levels when dealing with tweets as analyzed in (Liu, 2012). Bag-of-words models and naive Bayes models are sequence-agnostic, and have therefore failed to generalize over a diverse distribution of sentiments, especially when multiple fine-grained emotions are compressed into a 140 character message. Word vectors trained on a large corpus to represent the word in dense representations have

[1] https://twitter.com/i/moments/851423833160634368

Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pages 299–308

https://doi.org/10.18653/v1/P17

proved to be efficient in handling sentiment analysis and effective emotions. Deep learning and specifically Recurrent Neural Networks have been extensively used with word vectors to achieve state of the art results on various sentiment analysis tasks. Although there are large datasets available on social media space, deep learning models require annotated data for supervised training. Annotation for such a large dataset is expensive, since multiple human annotators are required per sample for stable convergence.

A useful research question is how to leverage resources available on social media sites to improve sentiment classification across datasets by leveraging the generic representations and handling the noise present in the space. These challenges have led people to use transfer learning and multi-task learning approaches to transfer knowledge across different datasets and languages. Recently, neural-network-based models for multi-task learning have become very popular, ranging from computer vision (Misra et al., 2016; Nam and Han, 2016) to natural language processing (Collobert et al., 2011a; Luong et al., 2015), since they provide a convenient way of combining information from multiple tasks.

We propose a dual Attention based deep learning model which creates representations using Bidirectional LSTM. In particular, given an input tweet, our model first uses a pair of bidirectional LSTMs to learn a general representation. This portion of the model is trained in a multi-task framework. The general sentence representation is then mapped into a task-specific representation through an attention mechanism, so that the most salient parts of the input are selected for each task. We achieve significant improvement over the baselines and obtain comparable results with the state of the art methods without any feature engineering.

We have selected datasets which classify a text into 3 classes, along with affect dataset. Affective dimensions provide much more granular analysis over emotions that are being conveyed. Affective emotions are classified along the valence, arousal and dominance axis according to circumplex model of affect, a well-established system for describing emotional states (Russell, 1980; Posner et al., 2005). Of these states, valence can directly be mapped to sentiment classification. These scales represent valence (or sentiment) and arousal (or intensity), which defines each posts position on the circumplex of the 3 dimension

The major contributions of this paper are:

- Generating robust representation of a tweet from three different set of pre-trained embeddings which can handle emoji/smileys and out-of-vocabulary words in the dataset.
- Multi Task learning frame work using Bidirectional Long short Memory Networks (BiLSTM) and attention mechanism to effectively learn the representations across datasets.

We evaluate the effectiveness of the model with respect to both internal and external distribution. The former refers to the setting where distribution of the test data falls in one of the m training tasks, and the latter refers to the setting where task and data are different and we use just the representation to train the task-specific layers.

Rest of the paper is organized as follows, section 2 discusses works related to multi-task learning along the lines of sentiment analysis. We present our proposed approach in section 3, which details the system architecture and its key components. Two sets of experiments and results shown in section 4 and 5 respectively. Finally section 6 concludes the paper with future direction.

2 Related Work

The current state of the art models for classifying the sentiments over social media text specifically tweets use a mixture of handcrafted features and pre-trained embeddings. Lexicon-based features along with neural network models to predict intensity of emotions have been proposed (Mohammad et al., 2013; Wilson et al., 2005; Ding et al., 2008; Bravo-Marquez et al., 2016; Esuli and Sebastiani, 2007) which have proved successful. However, these representations do not generalize well when there is a change in the vocabulary and the distribution. In addition, refining and generating handcrafted features is an expensive and tedious process. Our model do not require any hand-crafted features and can work with raw text and hence it can generalize well.

Two most popular embeddings that are being used are word2vec (Mikolov et al., 2013) and Glove (Pennington et al., 2014). Although these embeddings have improved the baselines from the

traditional bag-of-words model, they have been trained over large corpus in an unsupervised manner, they do not encode any sentiment information in them. The words like good and bad, due to their similar usage in the text appear to be close in the embedding space. To better represent the sentiment in the embeddings, several approaches to refine and learn embeddings have been reported. Learning of sentiment specific word embedding (SSWE) is presented in (Tang et al., 2014) where, embeddings were learned from a large corpus by incorporating the sentiment signal in the loss function. These embeddings are then used with different classifier such as convolutional Neural networks (CNN) followed by max-pooling (Collobert et al., 2011b; Socher et al., 2011; Mitchell and Lapata, 2010). We have considered this as one of our baselines.

Enriching of embedding using the distant-supervised method to learn set of embeddings using standard word2vec (Mikolov et al., 2013) and GLove (Pennington et al., 2014) is shown in (Deriu et al., 2016). Although this enrichment of embeddings is done using a large corpus of tweets, the basic assumption is that positive emoticons and emoji relate to the overall positive sentiment of the tweets create a lot of instability in the embedding space (Kunneman et al., 2014). This is due to the fact that, emoticons and emoji are used in various context and quite often in a polar opposite way to express sarcasm and irony (Poria et al., 2014). In addition, these methods are inefficient for more granular and fine-grained sentiment analysis.

3 Proposed Framework

In Figure 1 we present our generalized system diagram. Raw text is first preprocessed to normalize noise using standard text processing techniques.

3.1 Pre-processing

Tweets are essentially short text messages that are generated by humans to express their sentiments and reviews, and are known to be inherently noisy due to their condensed nature. This poses a challenge when trying to understand sentiment and affect. We have used standard text processing techniques with some modification to better suit the sentiment and affect domain:

- All the letters are converted to lower case form
- Significant amount of words are elongated with repeated number of characters such as "ANGRYYYYYYYYYY", we have limited these consequent characters to maximum of 2
- All the hyperlinks are removed as they do not serve the sentiment that is conveyed by the text itself and might relate to the sentiment pointed out by that links
- For words represented in hastags we remove "#" symbol, and if the word is not found in the vocabulary we try to segment it using Viterbi algorithm (Segaran and Hammerbacher, 2009)
- Usernames are replaced with "mention" token
- Compacted versions of word phrases such as "wasn't", "when's", etc., are replaced with corresponding expanded words

3.2 Embedding Generation

Processed text is then used to generate two sets of embeddings. First set of embeddings are generated by using three different pre-trained embeddings.

- Pre-trained embeddings which are generated from common crawl corpus have 6 Billion tokens which help in a better encoding of the syntactic and semantic structure of the language.
- Pre-trained emoji (Eisner et al., 2016) embeddings are used to represent the emojis and emoticons in the text. Emojis and emoticons are essential part of text which strongly convey the sentiments.
- To handle out of vocabulary words after the segmentations and the spelling corrections, we use character embedding[2] to generate a representations by summing all the character embeddings in that word. This helps in capturing sentiment related signals better than assigning it to the random tokens.

Pseudo code 1 details the process of generating first set of embedding which uses Glove embeddings trained over common crawl corpus of vector

[2]https://github.com/minimaxir/char-embeddings

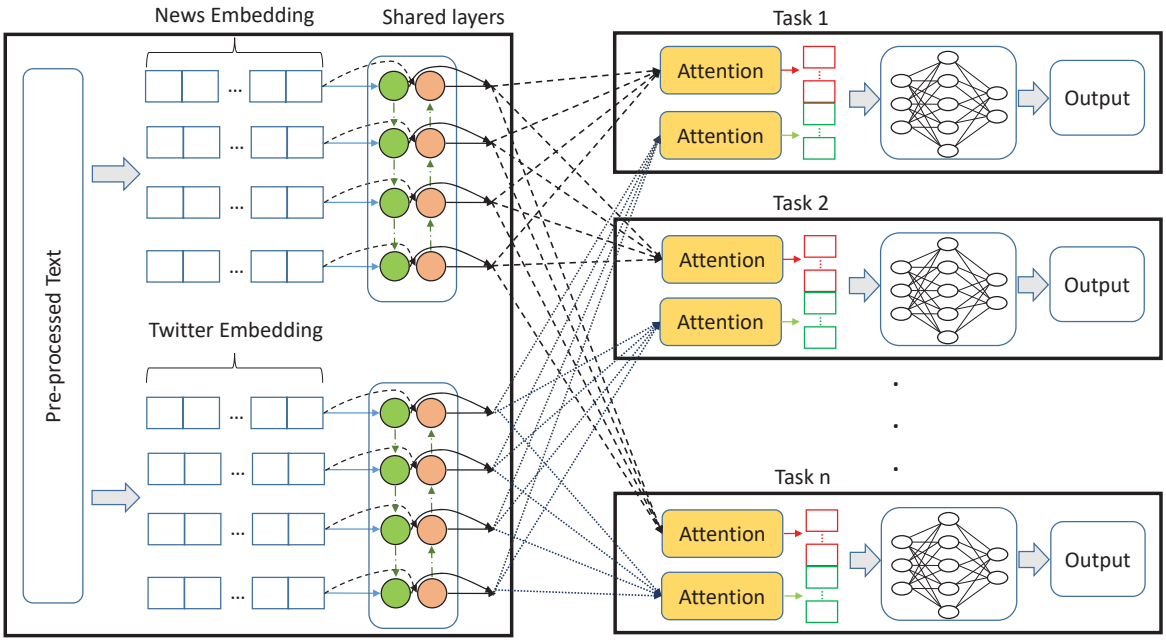

Figure 1: System Architecture

for each word. Second set of embeddings are extracted from pre-trained embeddings over Twitter corpus to get vectors that represents the nuances of the Twitter platform and in general of short and noisy text. These embeddings provide varied vector sizes, we use *300* and *200* dimensions of embedding for common crawl and twitter respectively.

word_token = Tokenize tweet
for *each word in word_token* **do**
  if *word is in EmojiEmbb* **then**
    word_vector = get_vector(EmojiEmbb, word_vector)
  else if *word is in Glove* **then**
    word_vector = get_vector(Glove, word_vector)
  else if *word is in CharEmbb* **then**
    word_vector = get_vector(charEmbb, word_vector)
  else
    chars = tokeinze word_token into character
    n = length(chars) word_vector = $\sum_1^n get\_vector(charEmbb, chars)$
  end
end

Algorithm 1: Embedding Matrix generation

Embeddings are then zero padded to match the sequence length across the datasets of different task. We have used *90* words as the maximum sequence length to account for any variations in validation datasets. For generalization, single sample of processed text can be represented in form of two sets of matrix as $\langle n_w \times d_g \rangle$ and $\langle n_w \times d_t \rangle$, where n_w is maximum sequence length or maximum number of words present in the text and d_g, d_t are the dimension of each embeddings. In this paper $n_w = 90$, $d_g = 300$ and $d_t = 200$.

These embeddings are then fed into 2 separate BiLSTM layers for each set of embeddings.

3.3 Model Description

We use LSTM architecture that was proposed in (Graves, 2013), which is governed by following equations,

$$
\begin{aligned}
i_t &= tanh(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \\
j_t &= sigm(W_{xj}x_t + W_{hj}h_{t-1} + b_j) \\
f_t &= sigm(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \\
o_t &= tanh(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \\
c_t &= c_{t-1} \oplus f_t + i_t \oplus j_t \\
h_t &= tanh(c_t) \oplus o_t
\end{aligned}
$$

In these equations, the W_* are the weight matrices and b_* are biases. The operation $\oplus$ denotes the element-wise vector product. The variable c_t denotes memory of LSTM at time step t.

Bidirectional Long Short Term memory (BiLSTMs) are improvement over LSTM networks where, two LSTM layers are stacked over each other. One of the layer processes the sequence in the forward pass and another process the seqeunce in backward fashion. Equations for the LSTM layers remain same and training can be done using stochastic gradient descent. So at each time step t, we receive set of h_t, one from forward pass and one from backward pass in BiLSTM, we concatenate and term it as a single output h_t.

As can be seen from above equation, forget gate bias can prove to be inefficient if it is initialized to random value and might introduce problem of vanishing gradient problem by a factor of 0.5 (Hochreiter and Schmidhuber, 1997; Martens and Sutskever, 2011). This can adversely affect the long term dependencies, to address this problem we initialize forget gate bias b_f to value just above 1 to facilitate the gradient flow as suggested in (Gers et al., 2000; Jozefowicz et al., 2015). To further regularize and avoid over-fitting dropout is used.

The output of both BiLSTM layers is then fed into the task-specific layers. Figure 1 shows the task-specific layers, where attention is used over the output of each BiLSTM layer. Attention was initially proposed for Neural Machine Translation (NMT) for encoder-decoder architecture to provide a context in terms of weights to important words (Bahdanau et al., 2014). In our problem where the final goal is to classify or to predict the intensity, attention is only required at the encoding level. Context vector can be computed using for the output of RNNs are follows,

$$c_t = \sum_{j=1}^{T} \alpha_{tj} h_j$$
$$e_t = a_(h_t), a_t = \frac{exp(e_t)}{\sum_{k=1}^{T} exp(e_k)}$$

where T is the total number of time steps in the sequence(in our case maximum sequence length) and α_{tj} is the weight computed for hidden state h_j at each time step t. Context vector c_t are then used to compute new sequence using previous state in the sequence and the context vectors. This ensures the new sequences have direct access to the entire sequence h.

Output of the attention layer is then fed to the fully connected layers, and the size and activation of the final layer depends on the task at hand.

4 Experiments

In order to validate our approach we perform two experiments. In experiment-1 we train our model on mixture of regression and classification tasks and access its performance over the same task by fine tuning it for the same task. In experiment-2 we accesses the representations that are obtained during experiment-1 on a different task.

4.1 Experiment-1: Multi-task training

We train and evaluate our model on sentiment classification SemEval dataset obtained through shared task and affect emotion dataset from SemEval-2018. These tasks are based on Twitter text and align to our objective of classifying short and noisy text present in the social media space. Although sentiment and affect task require a varying degree of representation where sentiment classification in positive, negative and neutral space can be relatively easier, representations required for this tasks are not present in the pre-trained embeddings.

4.1.1 Datasets

For sentiment classification dataset we use SemEval-2017 Task 4 Subtask A dataset. It contains a tweet and its respective label from positive, negative and neutral in english language. From here on we refer to this dataset as Sem-3. The classes presented are imbalance and negative tweets are around *15%* in training set and *32%* in test set (Rosenthal et al., 2017).

For Affective emotion, we use dataset which was presented as in SemEval-2018 task 1 (Mohammad et al., 2018) subtask *EI-reg, EI-oc* contains tweets specific to 4 emotions namely, Anger, fear, Joy and Sadness for english language. Subtask *V-reg, V-oc* contains the tweets for valence denoting range of positive to negative of sentiment. In subtasks *EI-reg* and *V-reg*, Given a tweet and its corresponding emotion, predict the intensity score of that emotion between 0 to 1, 0 being lowest and 1 being highest. Whereas, for subtasks *EI-oc* and *V-oc* we need to classify them into predefined classes, where oc means ordinal classification. In this dataset, emotions are classified in 4 distinct labels from mildly felt emotion to strongly felt emotion, while valence is classified into 7 distinct classes. Distribution of the datasets into the train development and test set is presented in table 1.

Table 1: Data Distribution.

	Train	Dev	Test
Anger	1701	388	1002
Fear	2252	389	986
Joy	1616	290	1105
Sadness	1533	397	975
Valence	1181	449	937
Sem-3	50334	20632	12284

Predicting intensity for emotions and valence are considered as regression task, while classifying into one of the classes is considered as classification task. We have 5 regression tasks and 6 classification tasks across these two datasets.

4.1.2 Training Procedure

The sem-3 dataset have approximately 15 time more training samples on an average when compared to all the rest of tasks assuming regression and classification are different tasks for each emotions. We define a training algorithm mentioned in algorithm 2. We train for sem-3 task for 1 epochs while others are trained for 15 epochs to account for the sample imbalances. We have chosen to keep the validation and test dataset as it is to better compare over the baselines.

for *episode in episodes* **do**
  train sem-3 for 1 epoch
  list = random order of task rest of 10 classes
  for *task in list* **do**
    train *task* for 15 epochs
  end
end

For classification tasks we have used categorical crossentropy as loss function, while for regression task we have defined a custom loss function as follows,

$$Loss = 0.7 \times (1 - pearson) + 0.3 \times MSE \quad (1)$$

where pearson is the pearson correlation and MSE is the mean squared error. As pearson correlation was the official metric for the regression task and has proven to be better representative than mean squared error. We have taken mean squared error into account to decrease the bias than creeps in due to batch size while training.

For classification tasks, class weights were applied in the loss function to handle class imbalances. Weights were set according to the inverse of their frequency.

Model hyper-parameters are shown in table 2. In addition, *0.5* and *0.35* dropout was used for fully connected layer and BiLSTM respectively. These parameters were chosen using grid search over validation dataset. We have used *Tanhyperbolic* as for BiLSTM and *Scaled Exponential Linear Units (selu)* (Klambauer et al., 2017) for fully connected layers as activation function. Fine tuning for each task is by freezing the shared layer weights after training to generate results for individual tasks.

Table 2: Details of layers

Layers	$Classification$	$Regression$
BiLSTM Layer 1	70	70
BiLSTM Layer 2	70	70
Fully connected layer 1	100	100
Fully connected layer 2	50	50
Fully connected layer 3	$3/5/7$	1

4.2 Experiment 2: Validating on external distribution

Irony detection in the social media is one such field which is correlated with the sentiment analysis. Although it requires different set of features, sentiment and affective emotions enhances the detection accuracies as reported in (Farías et al., 2016; Wallace, 2015). In this experiment, we apply representations generated earlier to irony classification to access its robustness. We have used irony detection dataset introduced in SemEval-2018 task 3 (Van Hee et al., 2018). Dataset was augmented and hashtags used to mine the tweets such as "#irony", "#sarcasm", etc., were omitted for testing. We have removed this hashtags during training as well to keep the dataset consistent.

This task contained two subtask, namely Subtask A and Subtask B. Objective of Subtask A was to classify whether a tweet contains irony or not, while of Subtask B was to classify into verbal irony (V-irony), situational irony (S-irony), other types of irony (O-irony) and non-ironic. Distribution of the dataset along the training and testing is presented in table 3.

Table 3: Distribution of Irony Dataset across train and test

	Subtask A		Subtask B			
	Ironic	Non-Ironic	V-irony	S-irony	O-irony	Non-ironic
Train	1911	1923	1390	316	205	1923
Test	311	473	164	85	62	473

For this we extract the representations from the model trained in experiment 1, specifically we take output of 2 BiLSTM layers. So for each sample in this dataset we have a 2D matrix of shape $\langle n_w \times b1 \rangle$, n_w is the maximum sequence length and $b1$ is the number of hidden units in the BiLSTM layer 1. Similarly we obtain the representation from BiLSTM layer 2. We concatenate this representation and pass it on to classification network consisting of single BiLSTM layer and two fully connected layers.

5 Results and Discussions

We ran our multi-task experiment for 10 episodes which translates to 100 training rounds. Figure 2 shows loss vs timesteps graph. Graphs are plotted differently to account for the different loss function scales. We can clearly observe the convergence over the time steps across tasks.

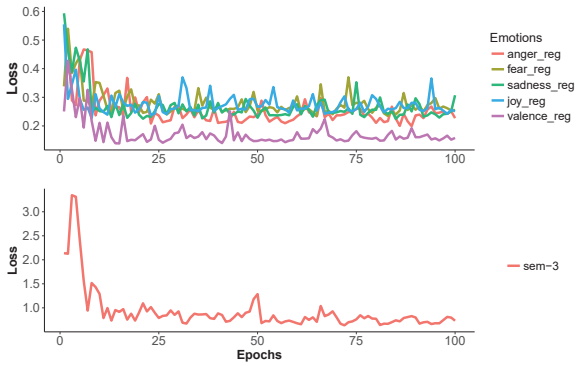

Figure 2: Upper graph shows the plot of loss vs episodes for regression task, while lower graph shows the corresponding plot for classification task for sem-3 task

For experiment 1, we use standard baselines as reported for the respective subtasks. In addition, we train a simple CNN classifier and LSTM classifier as our baselines. For *EI-reg, EI-oc, V-reg and V-oc*, baseline system was developed using wordvectors along with lexicons and support vector as final classification/regressor. We also compare the results with sentiment specific word embeddings (Tang et al., 2014), where we use Fully connected layers along with attention as the downstream model. For Sem-3 dataset we compare our results with RCNN (Yin et al., 2017) and Siamese network (Baziotis et al., 2017), which were top performing teams in the task. In addition, we separately train each task with same model parameters

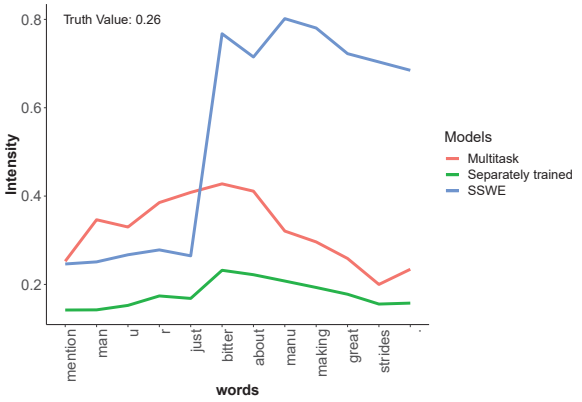

Figure 3: Plot of how intensity of the tweet changes with the words, signifying the importance of the sequence

without multi-task frame work, to observe the improvement due to multi-task and also to access the ability of the model architecture proposed. Results are shown in table 4, where for emotions and valence pearson correlation is reported and for sem-3 task accuracy and F1 score(F1 score is averaged for positive and negative class) is reported. Our model clearly out performs the baselines, and also provides significant increase over the recently proposed model architecture. Results also shows that separately trained model is able to beat the baselines, while adding the multi-task framework is able to boost the results further.

For experiment 2, baseline is unigram tf-idf features with Support vector classifiers. We compare it with the standard CNN and BiLSTM architectures. In addition, we also compare against recently proposed generalized representation for language modeling (Peters et al., 2018), which has been a state of the art for Yelp and IMDB dataset. These representations are available in two sets; a weighted sum of three layers of BiLSTMs (samples_size, max_length, 1024) refereed as ELMO-3D and fixed mean-pooling of all contextualized word representations (samples_size, 1024) refereed as ELMO-2D. For ELMO-3D embeddings we have used attention and fully connected layer as the classifier and for 2 dimension embeddings, we have used fully connected layers as classifiers. Results are shown in table 5, where F1-score is reported over the classes, for multiclass subtask average of F1 score for each class has been reported, this was the official metric of this task. We find that our framework out-performs all the baselines

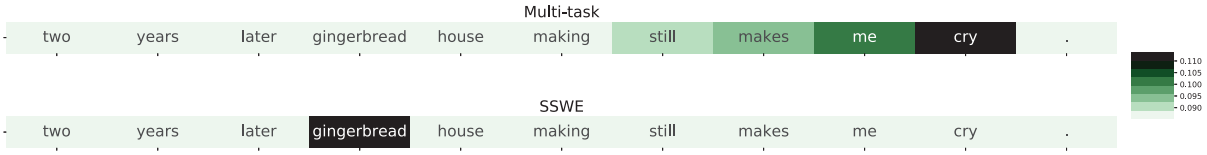

Figure 4: Attention weights comparison of Multi-task and SSWE

Table 4: Results of Experiment 1

	anger		fear		joy		sadness		valence		sem-3	
	reg	oc	reg	oc	reg	oc	reg	oc	reg	oc	Acc.	$F1_{PN}$
Baseline	0.526	0.382	0.525	0.355	0.575	0.469	0.453	0.370	0.585	0.509	0.333	0.162
CNN	0.556	0.445	0.579	0.462	0.601	0.534	0.573	0.459	0.714	0.591	0.545	0.556
BiLSTM	0.627	0.511	0.635	0.497	0.612	0.556	0.613	0.507	0.73	0.621	0.637	0.646
SSWE	0.641	0.498	0.637	0.483	0.655	0.60	0.623	0.539	0.784	0.634	0.639	0.645
RCNN Ensemble (Yin et al., 2017)	-	-	-	-	-	-	-	-	-	-	0.664	0.658
Siamese LSTM (Baziotis et al., 2017)	-	-	-	-	-	-	-	-	-	-	0.651	0.**677**
Separately trained	0.697	0.551	0.696	0.527	0.689	0.629	0.685	0.597	0.804	0.667	0.642	0.64
Multitask	**0.732**	**0.622**	**0.736**	**0.575**	**0.728**	**0.664**	**0.722**	**0.628**	**0.832**	**0.703**	**0.672**	0.670

reported. Results are averaged across *10* runs to reduce the variance.

Table 5: Irony Detection F1 score

	Subtask A	Subtask B
Baseline	0.585	0.327
CNN	0.535	0.329
BiLSTM	0.592	0.396
ELMO-2Dim	0.591	0.406
ELMO-3Dim	0.604	0.412
SSWE	0.557	0.361
Multitask Representation	**0.629**	**0.425**

A possible reason for the low performance of pre-trained SSWE might be narrow vocabulary. We have around 95K words in our vocabulary whereas, SSWE had 137052 words in its pre-trained vocabulary out of which only 33473 words were in overlap with the dataset vocabulary. Although the embeddings are refined for sentiment words, social media space often contains words which are not present in the formal dictionaries, here as our model was able to generate embeddings of out of vocabulary words using character embeddings. Figure 3 shows an example from fear emotion, where plot of how the final intensity of the sentence is changed over different model is shown. SSWE jumps on the word "bitter" as the word contains highly negative sentiment associated with it, whereas the true value is low for fear. Proposed model is able to normalize over the sequence as the jump is not that drastic. Figure 4 shows comparison of how our proposed model learns to put weights to the words as compared to SSWE model. We believe that adding the sentiment context over in the embedding through multi-task training aided in the Irony classification dataset.

6 Conclusion

In this paper, we present an approach for generating representations using sentiment and affect dataset in the multi-task framework. We present our deep learning based model with a dual attention over two sets of embedding space to capture more rich nuances of Twitter while still keeping the semantic and syntactic structure of language. In addition, we use emoji and character embeddings to help in getting better sentiment specific signals and to mitigate the effect of out of vocabulary problem. Our experiments over both internal and external distribution of data show the effectiveness of the representation. We observe that our model perform significantly better as compared to the baselines and the current state of the art methods for the tasks. Going further, it would be effective to devise an algorithm to modify these representations with minimum computation and still adapt to a different domain.

References

Cem Akkaya, Janyce Wiebe, and Rada Mihalcea. 2009. Subjectivity word sense disambiguation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 190–199. Association for Computational Linguistics.

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

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 6: Siamese lstm with attention for humorous text comparison. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 390–395, Vancouver, Canada. Association for Computational Linguistics.

Felipe Bravo-Marquez, Eibe Frank, Saif M Mohammad, and Bernhard Pfahringer. 2016. Determining word–emotion associations from tweets by multi-label classification. In *WI'16*, pages 536–539. IEEE Computer Society.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011a. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011b. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Jan Deriu, Maurice Gonzenbach, Fatih Uzdilli, Aurelien Lucchi, Valeria De Luca, and Martin Jaggi. 2016. Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision. In *SemEval@ NAACL-HLT*, pages 1124–1128.

Xiaowen Ding, Bing Liu, and Philip S Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 international conference on web search and data mining*, pages 231–240. ACM.

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bosnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description.

Andrea Esuli and Fabrizio Sebastiani. 2007. Sentiwordnet: A high-coverage lexical resource for opinion mining. *Evaluation*, pages 1–26.

Delia Irazú Hernańdez Farías, Viviana Patti, and Paolo Rosso. 2016. Irony detection in twitter: The role of affective content. *ACM Transactions on Internet Technology (TOIT)*, 16(3):19.

Felix A. Gers, Jürgen A. Schmidhuber, and Fred A. Cummins. 2000. Learning to forget: Continual prediction with lstm. *Neural Comput.*, 12(10):2451–2471.

A. Graves. 2013. Generating Sequences With Recurrent Neural Networks. *ArXiv e-prints*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 2342–2350.

G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter. 2017. Self-Normalizing Neural Networks. *ArXiv e-prints*.

FA Kunneman, CC Liebrecht, and APJ van den Bosch. 2014. The (un) predictability of emotional hashtags in twitter.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2015. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*.

James Martens and Ilya Sutskever. 2011. Learning recurrent neural networks with hessian-free optimization. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1033–1040.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-stitch networks for multi-task learning. *CoRR*, abs/1604.03539.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 task 1: Affect in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1–17. Association for Computational Linguistics.

Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *arXiv preprint arXiv:1308.6242*.

Hyeonseob Nam and Bohyung Han. 2016. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4293–4302.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics.

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.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR*, abs/1802.05365.

Soujanya Poria, Erik Cambria, Gregoire Winterstein, and Guang-Bin Huang. 2014. Sentic patterns: Dependency-based rules for concept-level sentiment analysis. *Knowledge-Based Systems*, 69:45–63.

Jonathan Posner, James A Russell, and Bradley S Peterson. 2005. The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology. *Development and psychopathology*, 17(3):715–734.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada. Association for Computational Linguistics.

James A Russell. 1980. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161.

Toby Segaran and Jeff Hammerbacher. 2009. *Beautiful data: the stories behind elegant data solutions*. " O'Reilly Media, Inc.".

Raksha Sharma and Pushpak Bhattacharyya. 2013. Detecting domain dedicated polar words. In *IJCNLP*, pages 661–666.

Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, pages 801–809.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. 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*, pages 1201–1211. Association for Computational Linguistics.

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)*, volume 1, pages 1555–1565.

Cynthia Van Hee, Els Lefever, and Veronique Hoste. 2018. Semeval-2018 task 3: Irony detection in english tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 39–50. Association for Computational Linguistics.

Byron C Wallace. 2015. Computational irony: A survey and new perspectives. *Artificial Intelligence Review*, 43(4):467–483.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 347–354. Association for Computational Linguistics.

Yichun Yin, Yangqiu Song, and Ming Zhang. 2017. Nnembs at semeval-2017 task 4: Neural twitter sentiment classification: a simple ensemble method with different embeddings. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 621–625, Vancouver, Canada. Association for Computational Linguistics.