

# Multitask Learning for Fine-Grained Twitter Sentiment Analysis

Georgios Balikas

Univ. Grenoble Alps, CNRS, Grenoble INP - LIG/Coffreo  
geompalik@hotmail.com

Simon Moura

Univ. Grenoble Alps, CNRS, Grenoble INP - LIG  
simon.moura@univ-grenoble-alpes.fr

Massih-Reza Amini

Univ. Grenoble Alps, CNRS, Grenoble INP - LJK  
massih-reza.amini@univ-grenoble-alpes.fr

Traditional sentiment analysis approaches tackle problems like ternary (3-category) and fine-grained (5-category) classification by learning the tasks separately. We argue that such classification tasks are correlated and we propose a multitask approach based on a recurrent neural network that benefits by jointly learning them. Our study demonstrates the potential of multitask models on this type of problems and improves the state-of-the-art results in the fine-grained sentiment classification problem.

*Keywords:* Text Mining; Sentiment Analysis; Deep Learning; Multitask Learning, Twitter Analysis; bidirectional LSTM; Text classification

## Introduction

Automatic classification of sentiment has mainly focused on categorizing tweets in either two (binary sentiment analysis) or three (ternary sentiment analysis) categories (Giachanou & Crestani, 2016). In this work we study the problem of fine-grained sentiment classification where tweets are classified according to a five-point scale ranging from *VeryNegative* to *VeryPositive*. To illustrate this, Table 1 presents examples of tweets associated with each of these categories. Five-point scales are widely adopted in review sites like Amazon and TripAdvisor, where a user’s sentiment is ordered with respect to its intensity. From a sentiment analysis perspective, this defines a classification problem with five categories. In particular, Sebastiani et al. (Martino, Gao, & Sebastiani, 2016) defined such classification problems whose categories are explicitly ordered to be ordinal classification problems. To account for the ordering of the categories, learners are penalized according to how far from the true class their predictions are.

Although considering different scales, the various settings of sentiment classification are related. First, one may use the same feature extraction and engineering approaches to represent the text spans such as word membership in lexicons, morpho-syntactic statistics like punctuation or elongated word counts (Balikas & Amini, 2016; Kiritchenko, Zhu, & Mohammad, 2014). Second, one would expect that knowledge from one task can be transferred to the others and this would benefit the performance. Knowing that a tweet is “Positive” in the ternary setting narrows the classification decision between the *VeryPositive* and *Positive* categories in the

fine-grained setting. From a research perspective this raises the question of whether and how one may benefit when tackling such related tasks and how one can transfer knowledge from one task to another during the training phase.

Our focus in this work is to exploit the relation between the sentiment classification settings and demonstrate the benefits stemming from combining them. To this end, we propose to formulate the different classification problems as a multitask learning problem and jointly learn them. Multitask learning (Caruana, 1997) has shown great potential in various domains and its benefits have been empirically validated (Collobert & Weston, 2008; Plank, 2016; Liu, Qiu, & Huang, 2016b, 2016a) using different types of data and learning approaches. An important benefit of multitask learning is that it provides an elegant way to access resources developed for similar tasks. By jointly learning correlated tasks, the amount of usable data increases. For instance, while for ternary classification one can label data using distant supervision with emoticons (Go, Bhayani, & Huang, 2009), there is no straightforward way to do so for the fine-grained problem. However, the latter can benefit indirectly, if the ternary and fine-grained tasks are learned jointly.

The research question that the paper attempts to answer is the following: Can twitter sentiment classification problems, and fine-grained sentiment classification in particular, benefit from multitask learning? To answer the question, the paper brings the following two main contributions: (i) we show how jointly learning the ternary and fine-grained sentiment classification problems in a multitask setting improves the

VeryNegative	Beyond frustrated with my #Xbox360 right now, and that as of June, @Microsoft doesn't support it. Gotta find someone else to fix the drive.
Negative	@Microsoft Heard you are a software company. Why then is most of your software so bad that it has to be replaced by 3rd party apps?
Neutral	@ProfessorF @gilwuvsyou @Microsoft @LivioDeLaCruz We already knew the media march in ideological lockstep but it is nice of him to show it.
Positive	PAX Prime Thursday is overloaded for me with @Microsoft and Nintendo indie events going down. Also, cider!!! :p
VeryPositive	I traveled to Redmond today. I'm visiting with @Microsoft @SQLServer engineers tomorrow - at their invitation. Feeling excited.

Table 1

*The example demonstrates the different levels of sentiment a tweet may convey. Also, note the Twitter-specific use of language and symbols.*

state-of-the-art performance,<sup>1</sup> and (ii) we demonstrate that recurrent neural networks outperform models previously proposed without access to huge corpora while being flexible to incorporate different sources of data.

### Multitask Learning for Twitter Sentiment Classification

In his work, Caruana (Caruana, 1997) proposed a multitask approach in which a learner takes advantage of the multiplicity of interdependent tasks while jointly learning them. The intuition is that if the tasks are correlated, the learner can learn a model jointly for them while taking into account the shared information which is expected to improve its generalization ability. People express their opinions online on various subjects (events, products..), on several languages and in several styles (tweets, paragraph-sized reviews..), and it is exactly this variety that motivates the multitask approaches. Specifically for Twitter for instance, the different settings of classification like binary, ternary and fine-grained are correlated since their difference lies in the sentiment granularity of the classes which increases while moving from binary to fine-grained problems.

There are two main decisions to be made in our approach: the learning algorithm, which learns a decision function, and the data representation. With respect to the former, neural networks are particularly suitable as one can design architectures with different properties and arbitrary complexity. Also, as training neural network usually relies on back-propagation of errors, one can have shared parts of the network trained by estimating errors on the joint tasks and others specialized for particular tasks. Concerning the data representation, it strongly depends on the data type available. For the task of sentiment classification of tweets with neural networks, distributed embeddings of words have shown great potential. Embeddings are defined as low-dimensional, dense representations of words that can be obtained in an

unsupervised fashion by training on large quantities of text (Pennington, Socher, & Manning, 2014).

Concerning the neural network architecture, we focus on Recurrent Neural Networks (RNNs) that are capable of modeling short-range and long-range dependencies like those exhibited in sequence data of arbitrary length like text. While in the traditional information retrieval paradigm such dependencies are captured using  $n$ -grams and skip-grams, RNNs learn to capture them automatically (Dyer, Ballesteros, Ling, Matthews, & Smith, 2015). To circumvent the problems with capturing long-range dependencies and preventing gradients from vanishing, the long short-term memory network (LSTM) was proposed (Hochreiter & Schmidhuber, 1997). In this work, we use an extended version of LSTM called bidirectional LSTM (biLSTM). While standard LSTMs access information only from the past (previous words), biLSTMs capture both past and future information effectively (Huang, Xu, & Yu, 2015; Dyer et al., 2015). They consist of two LSTM networks, for propagating text forward and backwards with the goal being to capture the dependencies better. Indeed, previous work on multitask learning showed the effectiveness of biLSTMs in a variety of problems: (Alonso & Plank, 2016) tackled sequence prediction, while (Plank, 2016) and (Kiperwasser & Goldberg, 2016) used biLSTMs for Named Entity Recognition and dependency parsing respectively.

Figure 1 presents the architecture we use for multitask learning. In the top-left of the figure a biLSTM network (enclosed by the dashed line) is fed with embeddings  $\{X_1, \dots, X_T\}$  that correspond to the  $T$  words of a tokenized tweet. Notice, as discussed above, the biLSTM consists of two LSTMs that are fed with the word sequence forward and backwards. On top of the biLSTM network one (or more) hidden layers  $H_1$  transform its output. The output of  $H_1$  is led to the softmax layers for the prediction step. There are  $N$  softmax layers and each is used for one of the  $N$  tasks of the multitask setting. In tasks such as sentiment classification, additional features like membership of words in sentiment lexicons or counts of elongated/capitalized words can be used to enrich the representation of tweets before the classification step (Kiritchenko et al., 2014). The lower part of the network illustrates how such sources of information can be incorporated to the process. A vector "Additional Features" for each tweet is transformed from the hidden layer(s)  $H_A$  and then is combined by concatenation with the transformed biLSTM output in the  $H_M$  layer.

### Experimental setup

Our goal is to demonstrate how multitask learning can be successfully applied on the task of sentiment classification

<sup>1</sup>An open implementation of the system for research purposes is available at <https://github.com/balikasg/sigir2017>.

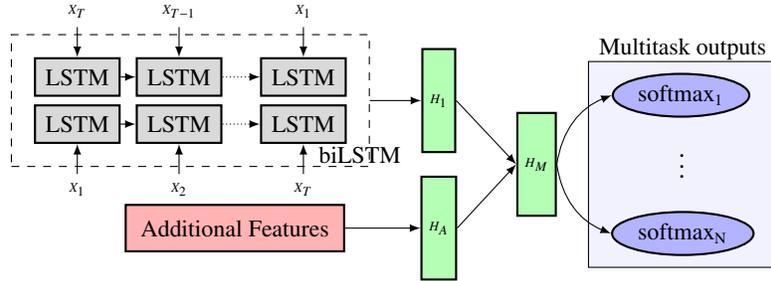


Figure 1. The neural network architecture for multitask learning. The biLSTM output is transformed by the hidden layers  $H_1$ ,  $H_M$  and is led to  $N$  output layers, one for each of the tasks. The lower part of the network can be used to incorporate additional information.

of tweets. The particularities of tweets are to be short and informal text spans. The common use of abbreviations, creative language etc., makes the sentiment classification problem challenging. To validate our hypothesis, that learning the tasks jointly can benefit the performance, we propose an experimental setting where there are data from two different twitter sentiment classification problems: a fine-grained and a ternary. We consider the fine-grained task to be our primary task as it is more challenging and obtaining bigger datasets, *e.g.* by distant supervision, is not straightforward and, hence we report the performance achieved for this task.

Ternary and fine-grained sentiment classification were part of the SemEval-2016 ‘‘Sentiment Analysis in Twitter’’ task (Nakov, Ritter, Rosenthal, Sebastiani, & Stoyanov, 2016). We use the high-quality datasets the challenge organizers released.<sup>2</sup> The dataset for fine-grained classification is split in training, development, development\_test and test parts. In the rest, we refer to these splits as `train`, `development` and `test`, where `train` is composed by the training and the development instances. Table 2 presents an overview of the data. As discussed in (Nakov et al., 2016) and illustrated in the Table, the fine-grained dataset is highly unbalanced and skewed towards the positive sentiment: only 13.6% of the training examples are labeled with one of the negative classes.

**Feature representation** We report results using two different feature sets. The first one, dubbed `nbow`, is a neural bag-of-words that uses text embeddings to generate low-dimensional, dense representations of the tweets. To construct the `nbow` representation, given the word embeddings dictionary where each word is associated with a vector, we apply the `average` compositional function that averages the embeddings of the words that compose a tweet. Simple compositional functions like `average` were shown to be robust and efficient in previous work (Mitchell & Lapata, 2010). Instead of training embeddings from scratch, we use the pre-trained on tweets GloVe embeddings of (Pennington et al., 2014).<sup>3</sup> In terms of resources required, using only `nbow` is efficient as it does not require any domain knowledge. However, previous research on sentiment analysis showed that us-

	$ D $	VeryNeg.	Neg.	Neutr.	Pos.	VeryPos.
Train	7,292	111	884	2,019	3,726	432
Dev.	1,778	29	204	533	887	125
Test	20,632	138	2,201	10,081	7,830	382
Ternary	5,500	-	785	1,887	2,828	-

Table 2

Cardinality and class distributions of the datasets.

ing extra resources, like sentiment lexicons, can benefit significantly the performance (Kiritchenko et al., 2014; Balikas & Amini, 2016). To validate this and examine at which extent neural networks and multitask learning benefit from such features we evaluate the models using an augmented version of `nbow`, dubbed `nbow+`. The feature space of the latter, is augmented using 1,368 extra features consisting mostly of counts of punctuation symbols (`!/?#@`), emoticons, elongated words and word membership features in several sentiment lexicons. Due to space limitations, for a complete presentation of these features, we refer the interested reader to (Balikas & Amini, 2016), whose open implementation we used to extract them.<sup>4</sup>

**Evaluation measure** To reproduce the setting of the SemEval challenges (Nakov et al., 2016), we optimize our systems using as primary measure the macro-averaged Mean Absolute Error ( $MAE_M$ ) given by:

$$MAE_M = \frac{1}{|C|} \sum_{j=1}^{|C|} \frac{1}{|Te_j|} \sum_{x_i \in Te_j} |h(x_i) - y_i|$$

where  $|C|$  is the number of categories,  $Te_j$  is the set of instances whose true class is  $c_j$ ,  $y_i$  is the true label of the instance  $x_i$  and  $h(x_i)$  the predicted label. The measure penalizes decisions far from the true ones and is macro-averaged to account for the fact that the data are unbalanced. Complementary to  $MAE_M$ , we report the performance achieved on

<sup>2</sup>The datasets are those of Subtasks A and C, available at <http://alt.qcri.org/semeval2016/task4/>.

<sup>3</sup>[urlhttp://nlp.stanford.edu/data/glove.twitter.27B.zip](http://nlp.stanford.edu/data/glove.twitter.27B.zip)

<sup>4</sup>[https://github.com/balikasg/SemEval2016-Twitter\\_Sentiment\\_Evaluation](https://github.com/balikasg/SemEval2016-Twitter_Sentiment_Evaluation)

the micro-averaged  $F_1$  measure, which is a commonly used measure for classification.

**The models** To evaluate the multitask learning approach, we compared it with several other models. Support Vector Machines (SVMs) are maximum margin classification algorithms that have been shown to achieve competitive performance in several text classification problems (Nakov et al., 2016).  $SVM_{OVR}$  stands for an SVM with linear kernel and an one-vs-rest approach for the multi-class problem. Also,  $SVM_{CS}$  is an SVM with linear kernel that employs the crammer-singer strategy (Crammer & Singer, 2001) for the multi-class problem. Logistic regression (LR) is another type of linear classification method, with probabilistic motivation. Again, we use two types of Logistic Regression depending on the multi-class strategy:  $LR_{OVR}$  that uses an one-vs-rest approach and multinomial Logistic Regression also known as the `MaxEnt` classifier that uses a multinomial criterion.

Both SVMs and LRs as discussed above treat the problem as a multi-class one, without considering the ordering of the classes. For these four models, we tuned the hyperparameter  $C$  that controls the importance of the  $L_2$  regularization part in the optimization problem with grid-search over  $\{10^{-4}, \dots, 10^4\}$  using 10-fold cross-validation in the union of the training and development data and then retrained the models with the selected values. Also, to account for the unbalanced classification problem we used class weights to penalize more the errors made on the rare classes. These weights were inversely proportional to the frequency of each class. For the four models we used the implementations of Scikit-learn (Pedregosa et al., 2011).

For multitask learning we use the architecture shown in Figure 1, which we implemented with Keras (Chollet, 2015). The embeddings are initialized with the 50-dimensional GloVe embeddings while the output of the biLSTM network is set to dimension 50. The activation function of the hidden layers is the hyperbolic tangent. The weights of the layers were initialized from a uniform distribution, scaled as described in (Glorot & Bengio, 2010). We used the Root Mean Square Propagation optimization method. We used dropout for regularizing the network. We trained the network using batches of 128 examples as follows: before selecting the batch, we perform a Bernoulli trial with probability  $p_M$  to select the task to train for. With probability  $p_M$  we pick a batch for the fine-grained sentiment classification problem, while with probability  $1 - p_M$  we pick a batch for the ternary problem. As shown in Figure 1, the error is backpropagated until the embeddings, that we fine-tune during the learning process. Notice also that the weights of the network until the layer  $H_M$  are shared and therefore affected by both tasks.

To tune the neural network hyper-parameters we used 5-fold cross validation. We tuned the probability  $p$  of dropout after the hidden layers  $H_M, H_1, H_A$  and for the biLSTM for  $p \in \{0.2, 0.3, 0.4, 0.5\}$ , the size of the hidden layer  $H_M \in$

$\{20, 30, 40, 50\}$  and the probability  $p_M$  of the Bernoulli trials from  $\{0.5, 0.6, 0.7, 0.8\}$ .<sup>5</sup> During training, we monitor the network’s performance on the development set and apply early stopping if the performance on the validation set does not improve for 5 consecutive epochs.

**Experimental results** Table 3 illustrates the performance of the models for the different data representations. The upper part of the Table summarizes the performance of the baselines. The entry “Balikas et al.” stands for the winning system of the 2016 edition of the challenge (Balikas & Amini, 2016), which to the best of our knowledge holds the state-of-the-art. Due to the stochasticity of training the biLSTM models, we repeat the experiment 10 times and report the average and the standard deviation of the performance achieved.

Several observations can be made from the table. First notice that, overall, the best performance is achieved by the neural network architecture that uses multitask learning. This entails that the system makes use of the available resources efficiently and improves the state-of-the-art performance. In conjunction with the fact that we found the optimal probability  $p_M = 0.5$ , this highlights the benefits of multitask learning over single task learning. Furthermore, as described above, the neural network-based models have only access to the training data as the development are hold for early stopping. On the other hand, the baseline systems were retrained on the union of the train and development sets. Hence, even with fewer resources available for training on the fine-grained problem, the neural networks outperform the baselines. We also highlight the positive effect of the additional features that previous research proposed. Adding the features both in the baselines and in the biLSTM-based architectures improves the  $MAE_M$  scores by several points.

Lastly, we compare the performance of the baseline systems with the performance of the state-of-the-art system of (Balikas & Amini, 2016). While (Balikas & Amini, 2016) uses n-grams (and character-grams) with  $n > 1$ , the baseline systems (SVMs, LRs) used in this work use the `nbow+` representation, that relies on unigrams. Although they perform on par, the competitive performance of `nbow` highlights the potential of distributed representations for short-text classification. Further, incorporating structure and distributed representations leads to the gains of the biLSTM network, in the multitask and single task setting.

Similar observations can be drawn from Figure 2 that presents the  $F_1$  scores. Again, the biLSTM network with multitask learning achieves the best performance. It is also to be noted that although the two evaluation measures are correlated in the sense that the ranking of the models is the same, small differences in the  $MAE_M$  have large effect on the scores of the  $F_1$  measure.

<sup>5</sup>Overall, we cross-validated 512 combinations of parameters. The best parameters were: 0.2 for all dropout rates, 20 neurons for  $H_M$  and  $p_M = 0.5$ .

	nbow	nbow+
SVM <sub>ovr</sub>	0.840	0.714
SVM <sub>cs</sub>	0.946	0.723
LR <sub>ovr</sub>	0.836	0.712
MaxEnt	0.842	0.715
(Balikas & Amini, 2016)	-	0.719
biLSTM (single task)	0.827±0.017	0.694±0.04
biLSTM+Multitask	0.786±0.025	<b>0.685±0.024</b>

Table 3

The scores on  $MAE_M$  for the systems. The best (lowest) score is shown in bold and is achieved in the multitask setting with the biLSTM architecture of Figure 1.

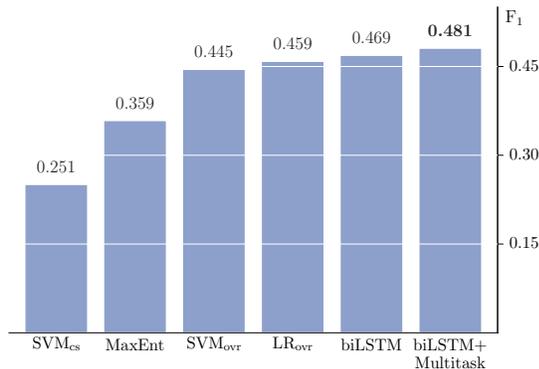


Figure 2.  $F_1$  scores using the nbow+ representations. The best performance is achieved with the multitask setting.

## Conclusion

In this paper, we showed that by jointly learning the tasks of ternary and fine-grained classification with a multitask learning model, one can greatly improve the performance on the second. This opens several avenues for future research. Since sentiment is expressed in different textual types like tweets and paragraph-sized reviews, in different languages (English, German, ..) and in different granularity levels (binary, ternary,..) one can imagine multitask approaches that could benefit from combining such resources. Also, while we opted for biLSTM networks here, one could use convolutional neural networks or even try to combine different types of networks and tasks to investigate the performance effect of multitask learning. Lastly, while our approach mainly relied on the foundations of (Caruana, 1997), the internal mechanisms and the theoretical guarantees of multitask learning remain to be better understood.

## Acknowledgements

This work is partially supported by the CIFRE N 28/2015.

## References

Alonso, H. M., & Plank, B. (2016). Multitask learning for semantic sequence prediction under varying data conditions.

*arXiv:1612.02251*.

- Balikas, G., & Amini, M. (2016). Twice at semeval-2016 task 4: Twitter sentiment classification. In *Semeval@naacl-hlt 2016* (pp. 85–91).
- Caruana, R. (1997). Multitask learning. *Machine Learning*, 28(1), 41–75.
- Chollet, F. (2015). *Keras*. <https://github.com/fchollet/keras>. GitHub.
- Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. In *icml*.
- Crammer, K., & Singer, Y. (2001). On the algorithmic implementation of multiclass kernel-based vector machines. *JMLR*, 2, 265–292.
- Dyer, C., Ballesteros, M., Ling, W., Matthews, A., & Smith, N. A. (2015). Transition-based dependency parsing with stack long short-term memory. In *Acl* (pp. 334–343).
- Giachanou, A., & Crestani, F. (2016). Like it or not: A survey of twitter sentiment analysis methods. *ACM Comput. Surv.*, 49(2), 28:1–28:41.
- Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Aistats* (pp. 249–256).
- Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1, 12.
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735–1780.
- Huang, Z., Xu, W., & Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.
- Kiperwasser, E., & Goldberg, Y. (2016). Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL*.
- Kiritchenko, S., Zhu, X., & Mohammad, S. M. (2014). Sentiment analysis of short informal texts. *JAIR*, 723–762.
- Liu, P., Qiu, X., & Huang, X. (2016a). Deep multi-task learning with shared memory for text classification. In *Emnlp* (pp. 118–127).
- Liu, P., Qiu, X., & Huang, X. (2016b). Recurrent neural network for text classification with multi-task learning. In *Ijcai* (pp. 2873–2879).
- Martino, G. D. S., Gao, W., & Sebastiani, F. (2016). Ordinal text quantification. In *Sigir* (pp. 937–940).
- Mitchell, J., & Lapata, M. (2010). Composition in distributional models of semantics. *Cognitive Science*, 34(8), 1388–1429.
- Nakov, P., Ritter, A., Rosenthal, S., Sebastiani, F., & Stoyanov, V. (2016). Semeval-2016 task 4: Sentiment analysis in twitter. In *Semeval@naacl-hlt* (pp. 1–18).
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *JMLR*, 12, 2825–2830.
- Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Emnlp* (pp. 1532–1543).
- Plank, B. (2016). Keystroke dynamics as signal for shallow syntactic parsing. In *Coling* (pp. 609–619).