# Multi-task Learning in Deep Neural Networks at EVALITA 2018

**Andrea Cimino**[⋆], **Lorenzo De Mattei**[⋆,◇] **and Felice Dell'Orletta**[⋆]
[⋆] Istituto di Linguistica Computazionale "Antonio Zampolli" (ILC–CNR)
ItaliaNLP Lab - *www.italianlp.it*
[◇] Dipartimento di Informatica, Università di Pisa
{andrea.cimino,felice.dellorletta}@ilc.cnr.it
lorenzo.dematteei@di.unipi.it

## Abstract

**English.** In this paper we describe the system used for the participation to the ABSITA, GxG, HaSpeeDe and IronITA shared tasks of the EVALITA 2018 conference. We developed a classifier that can be configured to use Bidirectional Long Short Term Memories and linear Support Vector Machines as learning algorithms. When using Bi-LSTMs we tested a multi-task learning approach which learns the optimized parameters of the network exploiting simultaneously all the annotated dataset labels and a multiclassifier voting approach based on a $k$-fold technique. In addition, we developed generic and specific word embedding lexicons to further improve classification performances. When evaluated on the official test sets, our system ranked 1st in almost all subtasks for each shared task, showing the effectiveness of our approach.

**Italiano.** *In questo articolo descriviamo il sistema utilizzato per la partecipazione agli shared task ABSITA, GxG, HaSpeeDe ed IronITA della conferenza EVALITA 2018. Abbiamo sviluppato un sistema che utilizza come algoritmi di apprendimento sia reti di tipo Long Short Term Memory Bidirezionali (Bi-LSTM) che Support Vector Machines. Nell'utilizzo delle Bi-LSTM abbiamo testato un approccio di tipo multi task learning nel quale i parametri della rete vengono ottimizzati utilizzando contemporaneamente le annotazioni presenti nel dataset ed una strategia di classificazione a voti di tipo $k$-fold. Abbiamo creato word embeddings generici e specifici per ogni singolo task per migliorare ulteriormente le performance di classificazione. Il nostro sistema quando valutato sui test set ufficiali ha ottenuto il primo posto in quasi tutti i sotto task di ogni shared task affrontato, dimostrando la validità del nostro approccio.*

## 1 Description of the System

The EVALITA 2018 edition has been one of the most successful editions in terms of number of shared tasks proposed. In particular, a large part of the tasks proposed by the organizers can be tackled as binary document classification tasks. This gave us the possibility to test a new system specifically designed for this EVALITA edition.

We implemented a system which relies on Bi-LSTM (Hochreiter et al., 1997) and SVM which are widely used learning algorithms in the document classification task. The learning algorithm can be selected in a configuration file. In this work we used the Keras (Chollet, 2016) library and the liblinear (Fan et al., 2008) library to generate the Bi-LSTM and SVM statistical models respectively. Since our approach relies on morphosyntactically tagged text, training and test data were automatically morphosyntactically tagged by the PoS tagger described in (Cimino and Dell'Orletta, 2016). Due to the label constraints in the dataset, if our system classified an aspect as not present, we forced the related positive and negative labels to be classified as not positive and not negative. We developed sentiment polarity and word embedding lexicons with the aim of improving the overall accuracy of our system.

Some specific adaptions were made due to the characteristics of each shared task. In the Aspect-based Sentiment Analysis (ABSITA) 2018 shared task (Basile et al., 2018) participants were asked, given a training set of Booking hotel reviews, to detect the mentioned aspect categories in a review among a set of 8 fixed categories (ACD task) and

to assign the polarity (neutral, positive, neutral, positive-negative) for each detected aspect (ACP task). Since each Booking review in the training set is labeled with 24 binary labels (8 indicating the presence of an aspect, 8 indicating positivity and 8 indicating negativity w.r.t. an aspect), we addressed the ABISTA 2018 shared task as 24 binary classification problems.

The Gender X-Genre (GxG) 2018 shared task (Dell'Orletta and Nissim, 2018) consisted in the automatic identification of the gender of the author of a text (Female or Male). Five different training sets and test sets were provided by the organizers for five different genres: Children essays (CH), Diary (DI), Journalism (JO), Twitter posts (TW) and YouTube comments (YT). For each test set the participants are requested to submit a system trained using in-domain training dataset and a system trained using cross-domain data only.

The IronITA task (Cignarella et al., 2018) consisted of two tasks. In the first task participants had to automatically label a message as ironic or not. The second task had a more fine grain: given a message, participants had to classify whether the message is sarcastic, ironic but not sarcastic or not ironic.

Finally in the HaSpeeDe 2018 shared task (Bosco et al., 2018) consisted in automatically annotating messages from Twitter and Facebook with a boolean value indicating the presence (or not) of hate speech. In particular three tasks were proposed: **HaSpeeDe-FB** where only the Facebook dataset could be used to classify Facebook comments, **HaSpeeDe-TW** where just Twitter data could be used to classify tweets and **Cross-HaspeeDe** where only the Facebook dataset could be used to classify the Twitter test set and vice versa (**Cross-HaspeeDe_FB**, **Cross-HaspeeDe_TW**).

## 1.1 Lexical Resources

### 1.1.1 Automatically Generated Sentiment Polarity Lexicons for Social Media

For the purpose of modeling the word usage in generic, positive and negative contexts of social media texts, we developed three lexicons which we named $TW_{GEN}, TW_{NEG}, TW_{POS}$. Each lexicon reports the relative frequency of a word in three different corpora. The main idea behind building these lexicons is that positive and negative words should present a higher relative frequency in $TW_{POS}$ and $TW_{NEG}$ respectively. The three corpora were generated by first downloading approximately 50,000,000 tweets and then applying some filtering rules to the downloaded tweets to build the positive and negative corpora (no filtering rules were applied to build the generic corpus). In order to build a corpus of positive tweets, we constrained the downloaded tweets to contain at least one positive emoji among heart and kisses. Since emojis are rarely used in negative tweets, to build the negative tweets corpus we created a list of commonly used words in negative language and constrained these tweets to contain at least one of these words.

### 1.1.2 Automatically translated Sentiment Polarity Lexicons

The Multi–Perspective Question Answering (hereafter referred to as $MPQA$) Subjectivity Lexicon (Wilson et al., 2005). This lexicon consists of approximately 8,200 English words with their associated polarity. To use this resource for the Italian language, we translated all the entries through the Yandex translation service[1].

### 1.1.3 Word Embedding Lexicons

We generated four word embedding lexicons using the word2vec[2] toolkit (Mikolov et al., 2013). As recommended in (Mikolov et al., 2013), we used the CBOW model that learns to predict the word in the middle of a symmetric window based on the sum of the vector representations of the words in the window. For our experiments, we considered a context window of 5 words. The Word Embedding Lexicons starting from the following corpora which were tokenized and postagged by the PoS tagger for Twitter described in (Cimino and Dell'Orletta, 2016):

- The first lexicon was built using the itWaC corpus[3]. The itWaC corpus is a 2 billion word corpus constructed from the Web limiting the crawl to the .it domain and using medium-frequency words from the Repubblica corpus and basic Italian vocabulary lists as seeds.

- The second lexicon was built using the set of the 50,000,000 tweets we downloaded to build the sentiment polarity lexicons previously described in subsection 1.1.1

[1] http://api.yandex.com/translate/

[2] http://code.google.com/p/word2vec/

[3] http://wacky.sslmit.unibo.it/doku.php?id=corpora

- The third and the fourth lexicon were built using a corpus consisting of 538,835 Booking reviews scraped from the web. Since each review in the Booking site is split in a positive secion (indicated by a plus mark) and negative section (indicated by a minus mark), we split these reviews obtaining in 338,494 positive reviews and 200,341 negative reviews. Starting from the positive and the negative reviews, we finally obtained two different word embedding lexicons.

Each entry of the lexicons maps a pair (word, POS) to the associated word embedding, allowing to mitigate polisemy problems which can lead to poorer results in classification. In addition, both the corpora where preprocessed in order to 1) map each url to the word "URL" 2) distinguish between all uppercased words and non-uppercased words (eg.: "mai" vs "MAI"), since all uppercased words are usually used in negative contexts. Since each task has its own characteristics in terms of information that needs to be captured from the classifiers, we decided to use a subset of the word embeddings in each task. Table 1 sums up the word embeddings used in each shared task.

| Task | Booking | ITWAC | Twitter |
|---|---|---|---|
| ABSITA | ✓ | ✓ | ✗ |
| GxG | ✗ | ✓ | ✓ |
| HaSpeeDe | ✗ | ✓ | ✓ |
| IronITA | ✗ | ✓ | ✓ |

Table 1: Word embedding lexicons used by our system in each shared task (✓used; ✗not used).

## 1.2 The Classifier

The classifier we built for our participation to the tasks was designed with the aim of testing different learning algorithms and learning strategies. More specifically our classifier implements two workflows which allow testing SVM and recurrent neural networks as learning algorithms. In addition, when recurrent neural networks are chosen as learning algorithms, our classifier allows to perform neural network multi-task learning (MTL) using an external dataset in order to share knowledge between related tasks. We decided to test the MTL strategy since, as demonstrated in (De Mattei et al., 2018), it can improve the performance of the classifier on emotion recognition tasks. The benefits of this approach were investigated also by Søgaard and Goldberg (2016), which showed that MTL is appealing since it allows to incorporate previous knowledge about tasks hierarchy into neural networks architectures. Furthermore, Ruder et al. (2017) showed that MTL is useful to combine even loosely related tasks, letting the networks automatically learn the tasks hierarchy.

Both the workflows we implemented share a common pattern used in machine learning classifiers consisting of a document feature extraction and a learning phase based on the extracted features, but since SVM and Bi-LSTM take input 2-dimensional and 3-dimensional tensors respectively, a different feature extraction phase is involved for each considered algorithm. In addition, when the Bi-LSTM workflow is selected the classifier can take as input an extra file which will be used to exploit the MTL learning approach. Furthermore, when the Bi-LSTM workflow is selected, the classifier performs 5-fold training approach. More precisely we build 5 different models using different training and validation sets. These models are then exploited in the classification phase: the assigned labels are the ones that obtain the majority among all the models. The 5-fold approach strategy was chosen in order to generate a global model which should less be prone to overfitting or underfitting w.r.t. a single learned model.

### 1.2.1 The SVM classifier

The SVM classifier exploits a wide set of features ranging across different levels of linguistic description. With the exception of the *word embedding combination*, these features were already tested in our previous participation at the EVALITA 2016 SENTIPOLC edition (Cimino et al., 2016). The features are organised into three main categories: *raw and lexical text features*, *morpho-syntactic features* and *lexicon features*. Due to size constraints we report only the feature names.

**Raw and Lexical Text Features** number of tokens, character $n$-grams, word $n$-grams, lemma $n$-grams, repetition of $n$-grams chars, number of mentions, number of hashtags, punctuation.

**Morpho-syntactic Features** coarse grained Part-Of-Speech $n$-grams, Fine grained Part-Of-Speech $n$-grams, Coarse grained Part-Of-Speech distribution

**Lexicon features** Emoticons Presence, Lemma sentiment polarity $n$-grams, Polarity modifier, PMI score, sentiment polarity distribution, Most frequent sentiment polarity, Sentiment polarity in text sections, Word embeddings combination.

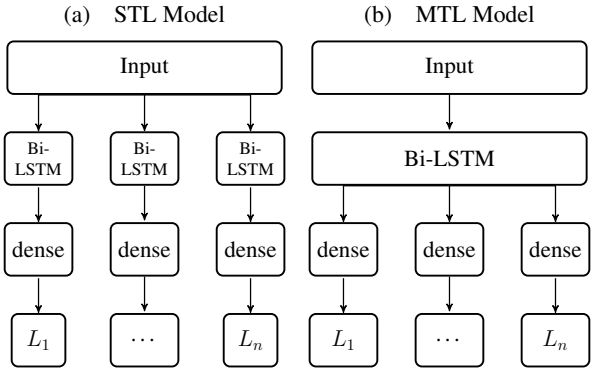
#### 1.2.2 The Deep Neural Network classifier

We tested two different models based on Bi-LSTM: one that learns to classify the labels without sharing information from all the labels in the training phase (Single task learning - STL), and the other one which learns to classify the labels exploiting the related information through a shared Bi-LSTM (Multi task learning - MTL). We employed Bi-LSTM architectures since these architectures allow to capture long-range dependencies from both directions of a document by constructing bidirectional links in the network (Schuster et al., 1997). We applied a dropout factor to both input gates and to the recurrent connections in order to prevent overfitting which is a typical issue in neural networks (Galp and Ghahramani, 2015). We have chosen a dropout factor value of 0.50.

For what concerns GxG, as we had to deal with longer documents such as news, we employed a two layer Bi-LSTM encoder. The first Bi-LSTM layer served us to encode each sentence as a token sequence, the second layer served us to encode the sentences sequence. For what concerns ironITA we added a task-specifici Bi-LSTM for each substask before the dense layer.

Figure 1 shows a graphical representation of the STL and MTL architectures we employed. For what concerns the optimization process, the binary cross entropy function is used as a loss function and optimization is performed by the rmsprop optimizer (Tieleman and Hinton, 2012).

Figure 1: STL and MTL architectures.




Each input word is represented by a vector which is composed by:

**Word embeddings**: the concatenation of the word embeddings extracted by the available Word Embedding Lexicons (128 dimensions for each word embedding), and for each word embedding an extra component was added to handle the "unknown word" (1 dimension for each lexicon used).

**Word polarity**: the corresponding word polarity obtained by exploiting the Sentiment Polarity Lexicons. This results in 3 components, one for each possible lexicon outcome (negative, neutral, positive) (3 dimensions). We assumed that a word not found in the lexicons has a neutral polarity.

**Automatically Generated Sentiment Polarity Lexicons for Social Media**: The presence or the absence of the word in a lexicon and the relative presence if the word is found in the lexicon. Since we built the $TW_{GEN}$, $TW_{POS}$ and $TW_{NEG}$ 6 dimensions are needed, 2 for each lexicon.

**Coarse Grained Part-of-Speech**: 13 dimensions.

**End of Sentence**: a component (1 dimension) indicating whether the sentence was totally read.

## 2 Results and Discussion

Table 2 reports the official results obtained by our best runs on all the task we participated. As it can be noted our system performed extremely well, achieving the best scores almost in every single subtask. In the following subsections a discussion of the results obtained in each task is provided.

### 2.1 ABSITA

We tested five learning configurations of our system based on linear SVM and DNN learning algorithms using the features described in section 1.2.1 and 1.2.2. All the experiments were aimed at testing the contribution in terms of f-score of MTL vs STL, the k-fold technique and the external resources. For what concerns the Bi-LSTM learning algorithm we tested Bi-LSTM both in the STL and MTL scenarios. In addition, to test the contribution of the Booking word embeddings, we created a configuration which uses a shallow Bi-LSTM in MTL setting without using these embeddings (MTL NO BOOKING-WE). Finally, to test the contribution of the k-fold technique we created a configuration which does not use the k-fold technique (MTL NO K-FOLD). To obtain fair comparisons in the last case we run all the experiments 5 times and averaged the scores of the runs. To test the proposed classification models, we created

| Task | Our Score | Best Score | Rank |
|---|---|---|---|
| ABSITA | | | |
| ACD | 0.811 | 0.811 | 1 |
| ACP | 0.767 | 0.767 | 1 |
| GxG IN-DOMAIN | | | |
| CH | 0.640 | 0.640 | 1 |
| DI | 0.676 | 0.676 | 1 |
| JO | 0.555 | 0.585 | 2 |
| TW | 0.595 | 0.595 | 1 |
| YT | 0.555 | 0.555 | 1 |
| GxG CROSS-DOMAIN | | | |
| CH | 0.640 | 0.640 | 1 |
| DI | 0.595 | 0.635 | 2 |
| JO | 0.510 | 0.515 | 2 |
| TW | 0.609 | 0.609 | 1 |
| YT | 0.513 | 0.513 | 1 |
| HaSpeeDe | | | |
| TW | 0.799 | 0.799 | 1 |
| FB | 0.829 | 0.829 | 1 |
| C_TW | 0.699 | 0.699 | 1 |
| C_FB | 0.607 | 0.654 | 5 |
| IronITA | | | |
| IRONY | 0.730 | 0.730 | 1 |
| SARCASM | 0.516 | 0.520 | 3 |

Table 2: Classification results of our best runs on the ABSITA, GxG, HaSpeeDe and IronITA test sets.

an internal development set by randomly selecting documents from the training sets distributed by the task organizers. The resulting development set is composed by approximately the 10% (561 documents) of the whole training set.

| Configuration | ACD | ACP |
|---|---|---|
| baseline | 0.313 | 0.197 |
| linear SVM | 0.797 | 0.739 |
| STL | 0.821 | 0.795 |
| MTL | **0.824** | **0.804** |
| MTL NO K-FOLD | 0.819 | 0.782 |
| MTL NO BOOKING-WE | 0.817 | 0.757 |

Table 3: Classification results (micro f-score) of the different learning models on our ABSITA development set.

Table 3 reports the overall accuracies achieved by the models on the internal development set for all the tasks. In addition, the results of baseline system (baseline row) which emits always the most probable label according to the label distribu-

| Configuration | ACD | ACP |
|---|---|---|
| baseline | 0.338 | 0.199 |
| linear SVM | 0.772* | 0.686* |
| STL | **0.814** | 0.765 |
| MTL | 0.811* | **0.767*** |
| MTL NO K-FOLD | 0.801 | 0.755 |
| MTL NO BOOKING-WE | 0.808 | 0.753 |

Table 4: Classification results (micro f-score) of the different learning models on the ABSITA official test set.

tions in the training set is reported. The accuracy is calculated as the micro f–score obtained using the evaluation tool provided by the organizers. For what concerns the ACD task it is worth noting that the models based on DNN always outperform linear SVM, even though the difference in terms of f-score is small (approximately 2 f-score points). The MTL configuration was the best performing among all the the models, but the difference in term of f-score among all the DNN configuration is not evident.

When analyzing the results obtained on the ACP task we can notice remarkable differences among the performances obtained by the models. Again the linear SVM was the worst performing model, but this time with a difference in terms of f-score of 6 points with respect to MTL, the best performing model on the task. It is interesting to notice that the results achieved by the DNN models have bigger difference between them in terms of f-score with respect to the ACD task: this suggests that the external resources and the k-fold technique contributed significantly to obtain the best result in the ACP task. The configuration that does not use the k-fold technique scored 2 f-score points w.r.t. the MTL configuration. We can also notice that the Booking word emebeddings were particularly helpful in this task: the MTL NO BOOOKING-WE configuration in fact scored 5 points less than the best configuration. The results obtained on the internal development set lead us to choose the models for the official runs on the provided test set. Table 4 reports the overall accuracies achieved by all our classifier configurations on the official test set, the official submitted runs are starred in the table.

As it can be noticed the best scores both in the ACD and ACP tasks were obtained by the DNN

models. Surprisingly the difference in terms of f-score were reduced in both the tasks, with the exception of linear SVM, which performed 4 and 8 f-score points less in the ACD and ACP tasks respectively when compared to the best DNN model systems. The STL model outperformed the MTL models the ACD task, even though the difference in term of f-score is not relevant. When the results on the ACP are considered, the MTL model outperformed all the other models, even though the the difference in terms of f-score with respect to the STL model is not noticeable. Is it worth to notice that the k-fold technique and the Booking word embeddings seemed to again contribute in the final accuracy of the MTL system. This can be seen by looking at the results achieved by the MTL NO BOOKING-WE model and the MTL NO K-FOLD model that scored 1.2 and 1.5 f-score points less than the MTL system.

### 2.2 GxG

We tested three different learning configurations of our system based on linear SVM and DNN learning algorithms using the features described in section 1.2.1 and 1.2.2. For what concerns the Bi-LSTM learning algorithm we tested both the STL and MTL approaches. We tested the three configurations for each of the 5 five in-domain subtasks and for each of the 5 five cross-domain subtasks. To test the proposed classification models, we created internal development sets by randomly selecting documents from the training sets distributed by the task organizers. The resulting development sets are composed by approximately 10% of the each data sets. For what concern the in-domain task, we tried to train the SVM classifier on in-domain-data only and and on both in-domain and cross-domain data.

| Model | CH | DI | JO | TW | YT |
|---|---|---|---|---|---|
| SVMa | 0.667 | 0.626 | 0.485 | 0.582 | 0.611 |
| SVM | **0.701** | 0.737 | 0.560 | 0.728 | 0.619 |
| STL | 0.556 | 0.545 | 0.500 | 0.724 | 0.596 |
| MTL | 0.499 | **0.817** | **0.625** | **0.729** | **0.632** |

Table 5: Classification results of the different learning models on development set in terms of accuracy for the **in-domain** tasks.

Table 5 and 6 report the overall accuracy, computed as the average accuracy for the two classes (male and female), achieved by the models on the development data sets for the in-domain and the cross-domain tasks respectively. For the in-domain tasks we observe that the SVM performs well on the smaller datasets (Children and Diary), while MTL neural network has the best overall performances. When trained on all the datasets, in- and cross-domain, the SVM (SVMa) perform worst than when trained on in-domain data only (SVM). For what concerns the cross-domain datasets we observe poor performances over all the subtasks with all the employed models, implying that the models have difficulties in cross-domain generalization.

| Model | CH | DI | JO | TW | YT |
|---|---|---|---|---|---|
| SVM | 0.530 | 0.565 | 0.580 | 0.588 | 0.568 |
| STL | 0.550 | 0.535 | 0.505 | 0.625 | 0.580 |
| MTL | 0.523 | 0.549 | 0.538 | 0.500 | 0.556 |

Table 6: Classification results of the different learning models on development set in terms of accuracy for the **cross-domain** tasks

| Model | CH | DI | JO | TW | YT |
|---|---|---|---|---|---|
| SVMa | 0.545 | 0.514 | 0.475 | 0.539 | **0.585** |
| SVM | 0.550 | 0.649 | **0.555** | 0.567 | 0.555* |
| STL | 0.545 | 0.541 | 0.500 | **0.595*** | 0.512 |
| MTL | **0.640*** | **0.676*** | 0.470 | 0.561 | 0.546 |

Table 7: Classification results of the different learning models on the official test set in terms of accuracy for the **in-domain** tasks (* marks runs that outperformed all the systems that participated to the task).

| Model | CH | DI | JO | TW | YT |
|---|---|---|---|---|---|
| SVM | 0.540 | 0.514 | **0.505** | 0.586 | **0.513*** |
| STL | **0.640*** | 0.554 | 0.495 | **0.609*** | 0.510 |
| MTL | 0.535 | **0.595** | 0.510 | 0.500 | 0.500 |

Table 8: Classification results of the different learning models on the official test set in terms of accuracy for the **cross-domain** tasks. (* marks runs that outperformed all the systems that participated to the task).

Table 7 and 8 report the overall accuracy, computed as the average accuracy for the two classes (male and female), achieved by the models on the official test sets for the in-domain and the cross-domain tasks respectively (* marks the running that obtain the best results in the competition). For what concerns the in-domain subtasks the perfor-

mances appear to be not in line with the ones obtained on the development set, but still our models outperform the other participant's systems in four out of five subtasks. The MTL model provided the best results for the Children and Diary test sets, while on the other test sets all the models performed quite poorly. Again when trained on all the datasets, in and cross-domain, the SVM (SVMa) perform worst then when trained on in-domain data only (SVM). For what concerns the cross-domain subtasks, while our model gets the best performances on three out of five subtasks, the results confirm poor performances over all the subtasks, again indicating that the models have difficulties in cross-domain generalization.

### 2.3 HaSpeeDe

We tested seven learning configurations of our system based on linear SVM and DNN learning algorithms using the features described in section 1.2.1 and 1.2.2. All the experiments were aimed at testing the contribution in terms of f-score of the number of layers, MTL vs STL, the k-fold technique and the external resources. For what concerns the Bi-LSTM learning algorithm we tested one and two layers Bi-LSTM both in the STL and MTL scenarios. In addition, to test the contribution of the sentiment lexicon features, we created a configuration which uses a 2-layer Bi-LSTM in MTL setting without using these features (1L MTL NO SNT). Finally, to test the contribution of the k-fold technique we created a configuration which does not use the k-fold technique (1 STL NO K-FOLD). To obtain fair results in the last case we run all the experiments 5 times and averaged the scores of the runs. To test the proposed classification models, we created two internal development sets, one for each dataset, by randomly selecting documents from the training sets distributed by the task organizers. The resulting development sets are composed by the 10% (300 documents) of the whole training sets.

Table 9 reports the overall accuracies achieved by the models on our internal development sets for all the tasks. In addition, the results of baseline system (baseline row) which emits always the most probable label according to the label distribution in the training set is reported. The accuracy is calculated as the f–score obtained using the evaluation tool provided by the organizers. For what concerns the Twitter in–domain task (TW

| Configuration | TW | FB | C_TW | C_FB |
|---|---|---|---|---|
| baseline | 0.378 | 0.345 | 0.345 | 0.378 |
| linear SVM | **0.800** | 0.813 | 0.617 | 0.503 |
| 1L STL | 0.774 | **0.860** | 0.683 | 0.647 |
| 2L STL | 0.790 | **0.860** | 0.672 | 0.597 |
| 1L MTL | 0.783 | **0.860** | 0.672 | **0.663** |
| 2L MTL | 0.796 | 0.853 | **0.710** | 0.613 |
| 1L MTL NO SNT | 0.793 | 0.857 | 0.651 | 0.661 |
| 1L STL NO K-FOLD | 0.771 | 0.846 | 0.657 | 0.646 |

Table 9: Classification results of the different learning models on our HaSpeeDe development set in terms of F1-score.

| Configuration | TW | FB | C_TW | C_FB |
|---|---|---|---|---|
| baseline | 0.403 | 0.404 | 0.404 | 0.403 |
| best official system | 0.799 | 0.829 | 0.699 | 0.654 |
| linear SVM | 0.798* | 0.761 | 0.658 | 0.451 |
| 1L STL | 0.793 | 0.811* | 0.669* | 0.607* |
| 2L STL | 0.791 | 0.812 | 0.644 | 0.561 |
| 1L MTL | 0.788 | 0.818 | **0.707** | **0.635** |
| 2L MTL | 0.799* | **0.829*** | 0.699* | 0.585* |
| 1L MTL NO SNT | **0.801** | 0.808 | 0.709 | 0.620 |
| 1L STL NO FOLD | 0.785 | 0.806 | 0.652 | 0.583 |

Table 10: Classification results of the different learning models on the official HaSpeeDe test set in terms of F1-score.

in the table) it is worth noting that linear SVM outperformed all the configurations based on Bi-LSTM. In addition, the MTL architecture results are slightly better than the STL ones (+1 f-score point with respect to the STL counterparts). External sentiment resources were not particularly helpful in this task, as shown by the result obtained by the 1L MTL NO SNT row. In the FB task, Bi-LSTMs sensibly outperformed linear SVMs (+5 f-score points in average); this is most probably due to longer text lengths that are found in this dataset with respect to the Twitter one. For what concerns the out–domain tasks, when testing models trained on Twitter and tested on Facebook (C_TW column), we can notice an expected drop in performance with respect to the models trained on the FB dataset (15-20 points f-score points). The best result was achieved by the 2L MTL configuration (+4 points w.r.t. the STL counterpart). Finally, when testing the models trained on Facebook and tested on Twitter (C_FB column), linear SVM showed a huge drop in terms of accuracy (-30 f-score points), while all the models trained with Bi-LSTM showed a performance drop of approximately 12 f-score points. Also in this

setting the best result was achieved by a MTL configuration (1L MTL), which performed better with respect to the STL counterpart (+2 f-score points). For what concerns the k-fold learning strategy, we can notice that the results achieved by the model not using the k-fold learning strategy (1 STL NO K-FOLD) are always lower than the counterpart which used the k-fold approach (+2.5 f-score points gained in the C_TW task), showing the benefits of using this technique.

These results lead us to choose the models for the official runs on the provided test set. Table 10 reports the overall accuracies achieved by all our classifier configurations on the official test set, the official submitted runs are starred in the table. The *best official system* row reports, for each task, the best official results submitted by the participants of the EVALITA 2018 HaSpeeDe shared task. As we can note the best scores in each task were obtained by the Bi-LSTM in the MTL setting, showing that MTL networks seem to be more effective with respect to STL networks. For what concerns the Twitter in–domain task, we obtained similar results to the development set ones. A sensible drop in performance is observed in the FB task w.r.t the development set (-5 f-score points in average). Still Bi-LSTMs models outperformed the linear SVM model by 5 f-score points. In the out-domain tasks, all the models performed similarly to what observed in the development set. It is worth observing that linear SVM performed almost as a baseline system in the C_FB task. In addition, in the same task the model exploiting the sentiment lexicon (1L MTL) showed a better performance (+1.5 f-score points) w.r.t to the 1L MTL NO SNT model. It is worth to notice that the k-fold learning strategy was beneficial also on the official test set: the 1L STL model obtained better results (approximately +2 f-score points in each task) w.r.t. the model that did not use the k-fold learning strategy.

### 2.4 IronITA

We tested the four designed learning configurations of our system based on linear SVM and deep neural network (DNN) learning algorithms using the features described in section 1.2.1 and 1.2.2. To select the proposed classification models, we used k-cross validation (k=4).

Table 11 reports the overall average f-score achieved by the models on the k-cross validation sets for both the irony and sarcarsm detection tasks.

| Configuration | Irony | Sarcasm |
|---|---|---|
| linear SVM | 0.734 | 0.512 |
| MTL | 0.745 | 0.530 |
| MTL+Polarity | 0.757 | **0.562** |
| MTL+Polarity+Hate | **0.760** | 0.557 |

Table 11: Classification results of the different learning models on k-cross validation terms of average F1-score.

| Configuration | Irony | Sarcasm |
|---|---|---|
| baseline-random | 0.505 | 0.337 |
| baseline-mfc | 0.334 | 0.223 |
| best participant | 0.730 | 0.52 |
| linear SVM | 0.701 | 0.493 |
| MTL | **0.736** | **0.530** |
| MTL+Polarity | 0.730* | 0.516* |
| MTL+Polarity+Hate | 0.713* | 0.503* |

Table 12: Classification results of the different learning models on the official test set in terms of F1-score (* submitted run).

We can observe that the SVM obtains good results on irony detection but the MTL neural approach overperforms sensibly the SVM. Also we note that the usage of additional Polarity and Hate Speech datasets lead to better performances. These results lead us to choose the MTL models trained with the additional dataset for the two official run submissions.

Table 12 reports the overall accuracies achieved by all our classifier configurations on the official test set, the official submitted runs are starred in the table. The accuracies has been computed in terms of F-Score using the official evaluation script. We submitted the runs MTL+Polarity and MTL+Polarity+Hate. The run MTL+Polarity ranked first in the subtask A, and third in the subtask B on the official leaderboard. The run MTL+Polarity ranked second in the subtask A, and fourth in the subtask B on the official leaderboard.

The results on the test set confirm the good performances of the SVM classifier on irony detection task and that the MTL neural approaches overperform the SVM. The model trained on the IronITA and SENTIPOLC datasets outperformed all the systems that participated to the subtask A, while on the subtask B it slightly underperformed the best participant system. The model trained on

the IronITA, SENTIPOLC and HaSpeeDe datasets overperformed all the systems that participated to the subtask A but our model trained on IronITA and SENTIPOLC datasets only. Although the best scores in both tasks were obtained by the MTL network trained on IronITA data set only. The MTL model trained on IronITA dataset only would have outperformed all the systems submitted to both the subtasks by all participants. Seems that for these tasks the usage of additional datasets leads to overfitting issues.

## 3 Conclusions

In this paper we reported the results of our participation to the ABSITA, GxG, HaSpeeDe and IronITA shared tasks of the EVALITA 2018 conference. By resorting to a system which used Support Vector Machines and Deep Neural Networks (DNN) as learning algorithms, we achieved the best scores almost in every task, showing the effectivness of our approach. In addition, when DNN was used as learning algorithm we introduced a new multi-task learning approach and a majority vote classification approach to further improve the overall accuracy of our system. The proposed system resulted in an very effective solution achieving the first position in almost all sub-tasks for each shared task.

## Acknowledgments

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 used for this research.

## References

Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli, Viviana Patti. 2016. Overview of the EVALITA 2016 SENTiment POLarity Classification Task. In *Proceedings of EVALITA '16, Evaluation of NLP and Speech Tools for Italian*. December, Naples, Italy.

Pierpaolo Basile, Valerio Basile, Danilo Croce, Marco Polignano. 2018. Overview of the EVALITA Aspect-based Sentiment Analysis (ABSITA) Task. T. Caselli, N. Novielli, V. Patti, P. Rosso, (eds). In *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*. December, Turin, Italy.

Cristina Bosco, Felice dell'Orletta, Fabio Poletto, Manuela Sanguinetti and Maurizio Tesconi. 2018. Overview of the Evalita 2018 Hate Speech Detection Task. T. Caselli, N. Novielli, V. Patti, P. Rosso, (eds). In *Proceedings of EVALITA '18, Evaluation of NLP and Speech Tools for Italian*. December, Turin, Italy.

Franois Chollet. 2016. Keras. *Software available at https://github.com/fchollet/keras/tree/master/keras*.

Alessandra Cignarella and Simona Frenda and Valerio Basile and Cristina Bosco and Viviana Patti and Paolo Rosso. 2018. Overview of the Evalita 2018 Task on Irony Detection in Italian Tweets (IronITA). T. Caselli, N. Novielli, V. Patti, P. Rosso, (eds). In *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*

Andrea Cimino and Felice dell'Orletta. 2016. Tandem LSTM-SVM Approach for Sentiment Analysis. In *Proceedings of EVALITA '16, Evaluation of NLP and Speech Tools for Italian*. December, Naples, Italy.

Andrea Cimino and Felice dell'Orletta. 2016. Building the state-of-the-art in POS tagging of Italian Tweets. In *Proceedings of Third Italian Conference on Computational Linguistics (CLiC-it 2016) & Fifth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2016), Napoli, Italy, December 5-7, 2016.*

Lorenzo de Mattei, Andrea Cimino and Felice dell'Orletta. 2018. Multi-Task Learning in Deep Neural Network for Sentiment Polarity and Irony classification. In *Proceedings of the 2nd Workshop on Natural Language for Artificial Intelligence, Trento, Italy, November 22-23, 2018.*

Felice Dell'Orletta and Malvina Nissim. 2018. Overview of the EVALITA Cross-Genre Gender Prediction in Italian (GxG) Task. T. Caselli, N. Novielli, V. Patti, P. Rosso, (eds). In *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang and Chih-Jen Lin 2008. LIBLINEAR: A Library for Large Linear Classification *Journal of Machine Learning Research.* Volume 9, 1871–1874

Yarin Gal and Zoubin Ghahramani. 2015. A theoretically grounded application of dropout in recurrent neural networks. *arXiv preprint arXiv:1512.05287*

Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short-term memory. *Neural computation*

Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv1:1301.3781.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani and Veselin Stoyanov. 2016. SemEval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein and Anders Søgaard. 2017. Document modeling with gated recurrent neural network for sentiment classification. arXiv preprint arXiv:1705.0814 1422-1432, Lisbon, Portugal.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45(11):2673–2681

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 231–235, Berlin, Portugal.

Duyu Tang, Bing Qin and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of EMNLP 2015*. 1422-1432, Lisbon, Portugal.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-RmsProp: Divide the gradient by a running average of its recent magnitude. In *COURSERA: Neural Networks for Machine Learning.*

Theresa Wilson, Zornitsa Kozareva, Preslav Nakov, Sara Rosenthal, Veselin Stoyanov and Alan Ritter. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of HLT-EMNLP 2005*. 347-354, Stroudsburg, PA, USA. ACL.

XingYi Xu, HuiZhi Liang and Timothy Baldwin. 2016. UNIMELB at SemEval-2016 Tasks 4A and 4B: An Ensemble of Neural Networks and a Word2Vec Based Model for Sentiment Classification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016).*