T100: A modern classic ensemble to profile irony and stereotype spreaders

Notebook for PAN at CLEF 2022

Marco Siino, Ilenia Tinnirello and Marco La Cascia

Università degli Studi di Palermo, Dipartimento di Ingegneria, Palermo, 90128, Italy

Abstract

In this work we propose a novel ensemble model based on deep learning and non-deep learning classifiers. The proposed model was developed by our team for participating at the Profiling Irony and Stereotype Spreaders (ISSs) task hosted at PAN@CLEF2022. Our ensemble (named T100), include a Logistic Regressor (LR) that classifies an author as ISS or not (nISS) considering the predictions provided by a first stage of classifiers. All these classifiers are able to reach state-of-the-art results on several text classification tasks. These classifiers (namely, *the voters*) are a Convolutional Neural Network (CNN), a Support Vector Machine (SVM), a Decision Tree (DT) and a Naive Bayes (NB) classifier. The voters are trained on the provided dataset and then generate predictions on the training set. Finally, the LR is trained on the predictions made by the voters. For the simulation phase the LR considers the predictions of the voters on the unlabelled test set to provide its final prediction on each sample. To develop and test our model we used a 5-fold cross validation on the labelled training set. Over the five validation splits, the proposed model achieves a maximum accuracy of 0.9342 and an average accuracy of 0.9158. As announced by the task organizers, the trained model presented here is able to reach an accuracy of 0.9444 on the unlabelled test set provided for the task.

Keywords

irony, stereotypes, author profiling, text classification, Twitter, ensemble, logistic regressor

1. Introduction

The task proposed at PAN@CLEF2022 [1] was about Profiling Irony and Stereotype Spreaders (ISSs) on Twitter [2]. The task was to investigate whether or not an author of a Twitter feed is likely to spread tweets containing irony and stereotypes. The organizers provided a labelled English dataset, consisting of 420 authors. In the dataset, each sample represents a single author's feed. For each author a set of 200 tweets is provided. The unlabelled test set provided consists of 180 samples. The model we used to compete for the task consists of a Logistic Regressor (LR) that get as input the predictions provided by a first stage of classifiers (named *the voters*). The voters are a Convolutional Neural Network (CNN), a Support Vector Machine (SVM), a Naive Bayes classifier (NB) and a Decision Tree (DT).

CLEF 2022: Conference and Labs of the Evaluation Forum, September 5–8, 2022, Bologna, Italy

marco.siino@unipa.it (M. Siino)

https://github.com/marco-siino (M. Siino)

0000-0002-4453-5352 (M. Siino); 0000-0002-1305-0248 (I. Tinnirello); 0000-0002-8766-6395 (M. L. Cascia)

Our paper is organized as follows. In Section 2 related works about deep and non-deep methods for text classification are presented. In Section 3 we describe our model (T100), including the training and the simulation steps. In Section 4 we discuss the experimental evaluation of our model, reporting the results of our tests on the 5-fold cross validation and on the test set. In Section 5 we propose future works and conclude the paper.

2. Related work

Recent approaches to the detection of stereotypes are proposed in [3, 4] while some interesting methods and discussions about irony detection are proposed in [5, 6]. However, to build up our model we investigated the best performing models participating at the shared tasks organized by PAN. Specifically, we looked at the last year author profiling task hosted at PAN@CLEF 2021, where the best performing model consisted of a shallow CNN presented in [7]. A previous edition of the author profiling task is discussed in [8], where the goal is to identify authors prone to spread fake news based on their last 100 tweets. The winners at the shared task were [9] and [10]. Their models obtained an overall accuracy of 0.77 on the provided test set. The approaches used by the winners are based on an SVM and n-grams and on an ensemble of different machine learning models.

Furthermore, we looked at common several state-of-the-art models on text classification tasks. It is worth reporting a significant increase in the use of Explainable Artificial Intelligence (XAI) methods in place of black box-based approaches. A few of these methods are based on graphs and used in real-world applications such as text classification [11], traffic prediction [12], computer vision [13] and social networking [14]. In [15] authors comparatively evaluate common machine learning algorithms (i.e., SVM, Naive Bayes, Logistic Regression and Recurrent Neural Networks (RNN)). On the dataset used, experimental results show that SVM and Naive Bayes outperform other methods. They do not report evaluation of CNN nor deep learning-based models in addition to the RNN. In another relevant comparative study [16], authors evaluate seven machine learning models on three different datasets. The models used are based on Random Forest, SVM, Gaussian Naive Bayes, AdaBoost, KNN, Multi-Layer Perceptron and Gradient Boosting Algorithm. In terms of accuracy and F1 score, the Gradient Boosting Algorithm outperforms the other tested models. However, also in this study, further experiments on deep models are missing.

In [17] the authors extend the CoAID dataset [18] to address the task of automatic detection of fake news spreaders of COVID-19 news. The authors present a stacked and Transformer-based neural network that combines the Transformer capabilities of computing sentence embeddings with a deep learning model. In [19], the authors use psycholinguistic and linguistic features as input to a CNN to profile fake news spreaders. The experimental results show that their proposed model is effective in classifying a user as a fake news spreader. The authors compare their results on a dataset specifically built for their task. However, the only Transformer tested is BERT and deep models performance is not widely explored. In addition, their proposed model is tested in [20] (where the PAN@CLEF2020 dataset is used) reporting poor results. Specifically, the model tested reaches a binary accuracy of 0.52 and of 0.51 on the English and Spanish dataset, respectively. In the same work [20], authors propose a new model that uses personality

information and visual features, outperforming the two winning models at PAN@CLEF2020 on both languages.

In the work conducted in [21], authors propose a CNN for the task of sentiment classification. Through experiments with three well-known datasets, authors show that employing consecutive convolutional layers is effective in classifying long texts.

Finally, the survey in [22] provides a brief overview of several text classification algorithms. This overview covers different text features extraction techniques, dimensionality reduction methods, existing algorithms and techniques, and evaluation methods.

Given the performances reached in a similar text classification task [23] and, as discussed in [24, 25], assuming that deep AI models are actually able to outperform classic techniques used in the field of natural language processing, we decided to include a deep learning-based model (i.e., a CNN) in our novel architecture.

The various and heterogeneous results of any of the state-of-the-art model discussed above, lead our team to develop an ensemble model able to classify a sample based on the predictions provided by a first stage of classifiers.

3. The proposed model: T100

The model proposed and described in this section is named T100. This name is motivated by the *modern classic* class of motorcycles produced by the UK-owned manufacturer[1]. In fact, T100 consists of both modern and classic elements to perform its task[2]. T100 include an LR model trained on the predictions provided by a first stage of classifiers. Details about the training phase of T100 are provided in the following subsection.

As a first step we preprocess each sample in our dataset to remove information common to all samples. More specifically we remove the tag CDATA before each tweet of any author's feed. Then we remove the starting tag <documents> opening each sample. Finally we remove the opening and closing tag <author lang="en">. Finally we lowercase all the text. The resulting text is then vectorized using the Keras Text Vectorization layer[3]. The preprocessing discussed above is performed by the text vectorization layer. Therefore, the text vectorization layer performs the following operations:

1. Preprocess the text of each sample
2. Split the text in each preprocessed sample into words (at each space character)
3. Recombine words into tokens (ngrams)
4. Index tokens (associate a unique int value with each token)
5. Transform each sample using this index, into a vector of ints.

While the vectorized text is provided as-is to the word embedding layer inside the CNN, another step is performed for other voters. The vectorized text is translated into a Bag-of-Words (BoW) representation and provided as input to the other voters (i.e., NB, SVM and DT).

[1] https://www.triumphmotorcycles.co.uk/

[2] ...that is text classification, not yet able to run at 100 MPH. Not yet...

[3] https://www.tensorflow.org/api_docs/python/tf/keras/layers/TextVectorization

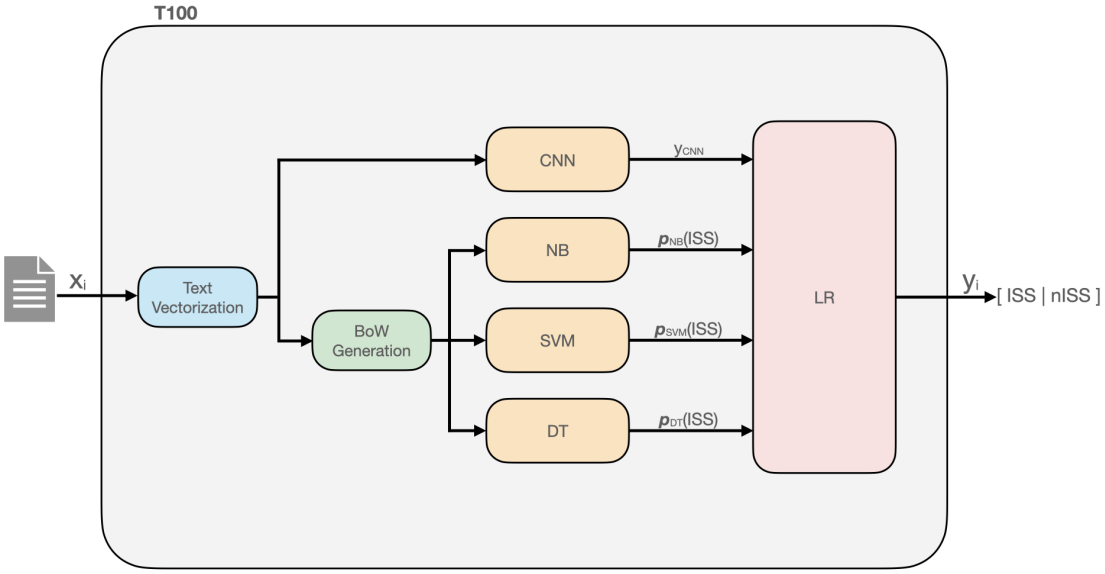

Figure 1: The overall architecture of T100. The sample $\mathbf{x}_i$ is the Twitter feed of the i-th author. The shallow CNN used in this work is built as discussed in [7]. Other classifiers are included into the *scikit-learn* package. LR uses the predictions provided by the voters to predict the label $\mathbf{y}_i$ corresponding to the input sample $\mathbf{x}_i$.

It is worth noting that the outputs from the first stage of classifiers have different meanings. In fact, the CNN outputs a float value in the range (-∞,+∞), while other classifiers output the probability that a given sample is an ISS. In the case of the CNN the threshold value is set equal to 0, therefore any negative value corresponds to a nISS while a positive one corresponds to an ISS.

The CNN network is implemented accordingly to the work discussed in [7] and in [23]. The network consists of a word embedding layer followed by a convolutional layer, an average pooling layer, a global average pooling layer and a single dense unit as output. The other voters are implemented using the *scikit-learn* packages[4].

At a very first implementation we tried to normalize each voter's output. Specifically we performed several experiments; as an instance, using the normalization techniques discussed in [26, 27]. However we discovered that keeping the original output range from each voter notably increases the performance of T100. So we lastly did not make use of any kind of normalization technique for any voter's output.

3.1. Model training

In this subsection we describe the training and the simulation phase of our novel architecture. The training of our model is based on a 5-fold strategy. As a first step we train each voter

[4]https://scikit-learn.org/stable/

using the k-training fold. Then we let each voter predicts on the corresponding k-validation fold. Then we merge the five sets of predictions on the validation folds. In such a way, a new predictions dataset is generated. In this new generated predictions dataset, samples consist of voter's predictions and of the original corresponding label (i.e., *nISS* or *ISS*) of the input sample. This new predictions dataset is used to train the LR.

After the training phase, the simulation phase is performed as follows. Using the official test set, we provide the unlabelled samples to the voters. Predictions of the voters are provided as input to the LR, then we collect and submit the final predictions made by the LR. This last prediction phase is depicted in Figure 1.

4. Experimental evaluation

Our model, developed in TensorFlow, is publicly available as a Jupyter Notebook on GitHub[5]. The architecture of the CNN-based model used in our work is very similar to the one discussed in [7]. It is a shallow CNN compiled with a binary cross entropy loss function; this function calculates loss with respect to two classes (i.e., 0 and 1). Optimization is performed with an Adamic optimizer [28] after giving each batch of data as input. For each fold we trained the CNN for five epochs. That is motivated by the fact that some overfitting starts after the fifth epoch. We performed a binary search to find the optimal batch size. The model achieved the best overall accuracy with a batch size equal to 1. For the NB voter we use *MultinomialNB* from the scikit-learn package. The SVM voter uses a linear kernel with a C-value equal to 0.5. Finally for the DT classifier we set a *random_state* equal to 0.

4.1. The dataset

The dataset provided by the PAN organizers consists of a set of 600 Twitter authors. For each author a set of 200 tweets is provided. A single XML file corresponds to an author and contains 200 tweets of the author. The labelled training set provided by the organizers contains 420 authors. The test set consists of the remaining 180 ones. Authors in the training set are labelled as "I" (ISS) or "NI" (nISS). Our final submission consists of a zip file containing predictions for each non-labelled author in the test set.

4.2. Results

The official metric used for the author profiling task at PAN@CLEF2022 is the accuracy. This metric is the same used in the rest of this section and defined in (1).

$$Accuracy = \frac{CorrectPredictions}{TotalPredictions} \tag{1}$$

Before performing the 5-fold cross validation we shuffled the 420 labelled samples and then we left out the last 40 samples as a labelled test set. In Table 1 are reported the results obtained by the single voters both on the test set and adopting a 5-fold cross validation on the labelled training set. In the table are reported the arithmetic mean and the standard deviation over

[5] https://github.com/marco-siino/T100-PAN2022

Voter	Set	Fold Nr.						
		1	2	3	4	5	AVG	σ
CNN	Val	0.8947	0.8684	0.9079	0.8684	0.8947	0.8868	0.0158
	Test	0.9000	0.8750	0.9250	0.9250	0.9500	0.9150	0.0255
NB	Val	0.8947	0.8553	0.8816	0.8289	0.8289	0.8579	0.0268
	Test	0.9000	0.9000	0.9000	0.8750	0.8750	0.8900	0.0122
SVM	Val	0.9210	0.9342	0.9079	0.8816	0.8947	0.9079	0.0186
	Test	0.8750	0.8500	0.8750	0.8750	0.8500	0.8650	0.0122
DT	Val	0.7368	0.8421	0.8684	0.7631	0.8816	0.8184	0.0579
	Test	0.7750	0.8000	0.7500	0.8500	0.8750	0.8100	0.0464

Table 1
Results in terms of accuracy achieved by each voter of T100 at each fold. Models are evaluated on the corresponding validation set at each fold and on the same test set. Performance of the classifiers at the first stage of T100 are lower compared to the ensemble model presented in this work. In the last two columns we report the values of the arithmetic mean and the standard deviation over the five folds.

T100 - Logistic Regressor	Fold Nr.						
	1	2	3	4	5	AVG	σ
Val	0.9210	0.9342	0.9342	0.8553	0.9342	0.9158	0.0307
Test	0.9250	0.9250	0.9250	0.9250	0.9250	0.9250	0.0000

Table 2
Results achieved by the model on a 5-fold cross validation on the training set provided. The results shown in the table are obtained using a Logistic Regressor as a final classifier of T100.

the 5-folds. Table 2 reports the results of T100 on the validation set at each fold and on the labelled 40 samples we used as a test set. In terms of accuracy, each classifier used individually performs worse than T100. Furthermore, standard deviation of the single voters and of T100 is comparable on the validation sets. However, the standard deviation is equal to 0 on the test set for T100 and higher for the single voters.

We performed several tests to investigate the best classifier as the very last predictor of T100. From Table 3 to Table 5 these results are reported.

How it is shown in the tables, the LR is consistent over different training fold, with a null standard deviation on the test set. In terms of consistency the Gradient Boosting Classifier performs similarly with a standard deviation of 0.010. However, results in term of binary accuracy are poor using Gradient Boosting Classifier as long as the other models tested.

Finally, we used the T100 trained at the fifth fold to generate the predictions on the official unlabelled test set provided by the organizers. As announced by the organizers, such a final version of our model is able to reach an accuracy of 0.9444 with respect to the official test set.

5. Conclusion and future works

In this paper we have described our submitted model for our participation at the Profiling ISSs on Twitter task at PAN 2022. It consists of an ensemble, T100, trained on the predictions of a first layer of classifiers. To get consistent evaluation of the model performance, we run several

T100 - Decision Tree	Fold Nr.						
	1	2	3	4	5	AVG	σ
Val	0.8421	0.8158	0.8947	0.8421	0.8158	0.8421	0.0288
Test	0.9000	0.8000	0.8500	0.8250	0.8500	0.8450	0.0331

Table 3
Results achieved by a T100 ensemble using a Decision Tree at the final prediction stage.

T100 - Random Forest	Fold Nr.						
	1	2	3	4	5	AVG	σ
Val	0.9079	0.9342	0.9210	0.8816	0.9210	0.9131	0.0178
Test	0.8750	0.9000	0.9000	0.8750	0.8750	0.8850	0.0122

Table 4
Results achieved by a T100 ensemble using a Random Forest at the final prediction stage.

T100 - Gradient Boosting	Fold Nr.						
	1	2	3	4	5	AVG	σ
Val	0.8816	0.9079	0.9210	0.8684	0.9210	0.9000	0.0214
Test	0.8750	0.8500	0.8500	0.8500	0.8500	0.8550	0.0100

Table 5
Results achieved by a T100 ensemble using a Gradient Boosting Classifier at the final prediction stage.

5-fold cross validations for each different hyperparameter configurations. After finding the model achieving the highest accuracy during our cross validation tests, we train such a model on the best train fold to submit our predictions on the unlabelled test set.

In future works, we expect to evaluate performance of our model increasing the number and the diversity of the voter classifiers employed at the first prediction stage. A detailed error analysis on misclassified samples could lead to improved performance on the classification task proposed. Given the dimension of the dataset provided some techniques of data augmentation could be also used. Finally, some investigation on the content of each tweet could guide us in applying some techniques to remove not relevant features from the input samples, before training and testing our proposed model.

Acknowledgments

We would like to thank anonymous reviewers for their comments and suggestions that have helped to improve the presentation of the paper.

CRediT Authorship Contribution Statement

Marco Siino: Conceptualization, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing - Original draft, Writing - review & editing. **Ilenia Tinnirello:** Writing - review & editing, Methodology. **Marco La Cascia:** Writing - review & editing, Methodology.

References

[1] J. Bevendorff, B. Chulvi, E. Fersini, A. Heini, M. Kestemont, K. Kredens, M. Mayerl, R. Ortega-Bueno, P. Pezik, M. Potthast, F. Rangel, P. Rosso, E. Stamatatos, B. Stein, M. Wiegmann, M. Wolska, E. Zangerle, Overview of PAN 2022: Authorship Verification, Profiling Irony and Stereotype Spreaders, and Style Change Detection, in: A. Barron-Cedeno, G. D. S. Martino, M. D. Esposti, F. Sebastiani, C. Macdonald, G. Pasi, A. Hanbury, M. Potthast, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Thirteenth International Conference of the CLEF Association (CLEF 2022), volume 13390 of *Lecture Notes in Computer Science*, Springer, 2022.

[2] O.-B. Reynier, C. Berta, R. Francisco, R. Paolo, F. Elisabetta, Profiling Irony and Stereotype Spreaders on Twitter (IROSTEREO) at PAN 2022, in: CLEF 2022 Labs and Workshops, Notebook Papers, CEUR-WS.org, 2022.

[3] J. Sánchez-Junquera, B. Chulvi, P. Rosso, S. P. Ponzetto, How do you speak about immigrants? taxonomy and stereoimmigrants dataset for identifying stereotypes about immigrants, Applied Sciences 11 (2021) 3610.

[4] J. Sánchez-Junquera, P. Rosso, M. Montes, B. Chulvi, et al., Masking and bert-based models for stereotype identication, Procesamiento del Lenguaje Natural 67 (2021) 83–94.

[5] S. Zhang, X. Zhang, J. Chan, P. Rosso, Irony detection via sentiment-based transfer learning, Information Processing & Management 56 (2019) 1633–1644.

[6] E. Sulis, D. I. H. Farías, P. Rosso, V. Patti, G. Ruffo, Figurative messages and affect in twitter: Differences between# irony,# sarcasm and# not, Knowledge-Based Systems 108 (2016) 132–143.

[7] M. Siino, E. Di Nuovo, I. Tinnirello, M. La Cascia, Detection of hate speech spreaders using convolutional neural networks, in: PAN 2021 Profiling Hate Speech Spreaders on Twitter@ CLEF, volume 2936, CEUR, 2021, pp. 2126–2136.

[8] F. Rangel, A. Giachanou, B. H. H. Ghanem, P. Rosso, Overview of the 8th author profiling task at pan 2020: Profiling fake news spreaders on twitter, in: CEUR Workshop Proceedings, volume 2696, Sun SITE Central Europe, 2020, pp. 1–18.

[9] J. Pizarro, Using n-grams to detect fake news spreaders on twitter, in: CLEF, 2020, p. 1.

[10] J. Buda, F. Bolonyai, An ensemble model using n-grams and statistical features to identify fake news spreaders on twitter, in: CLEF, 2020.

[11] F. Lomonaco, G. Donabauer, M. Siino, Courage at checkthat! 2022: Harmful tweet detection using graph neural networks and electra, in: Working Notes of CLEF 2022—Conference and Labs of the Evaluation Forum, CLEF '2022, Bologna, Italy, 2022.

[12] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, arXiv preprint arXiv:1707.01926 (2017).

[13] P. Pradhyumna, G. Shreya, et al., Graph neural network (gnn) in image and video understanding using deep learning for computer vision applications, in: 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC), IEEE, 2021, pp. 1183–1189.

[14] M. Siino, M. La Cascia, I. Tinnirello, Whosnext: Recommending twitter users to follow using a spreading activation network based approach, in: 2020 International Conference on Data Mining Workshops (ICDMW), IEEE, 2020, pp. 62–70.

[15] E. M. Mahir, S. Akhter, M. R. Huq, et al., Detecting fake news using machine learning and deep learning algorithms, in: 2019 7th International Conference on Smart Computing & Communications (ICSCC), IEEE, 2019, pp. 1–5.
[16] A. P. S. Bali, M. Fernandes, S. Choubey, M. Goel, Comparative performance of machine learning algorithms for fake news detection, in: International conference on advances in computing and data sciences, Springer, 2019, pp. 420–430.
[17] S. Leonardi, G. Rizzo, M. Morisio, Automated classification of fake news spreaders to break the misinformation chain, Information 12 (2021) 248.
[18] L. Cui, D. Lee, Coaid: Covid-19 healthcare misinformation dataset, arXiv preprint arXiv:2006.00885 (2020).
[19] A. Giachanou, B. Ghanem, E. A. Ríssola, P. Rosso, F. Crestani, D. Oberski, The impact of psycholinguistic patterns in discriminating between fake news spreaders and fact checkers, Data & Knowledge Engineering 138 (2022) 101960.
[20] R. Cervero, P. Rosso, G. Pasi, Profiling Fake News Spreaders: Personality and Visual Information Matter, in: International Conference on Applications of Natural Language to Information Systems, Springer, 2021, pp. 355–363.
[21] H. Kim, Y.-S. Jeong, Sentiment classification using convolutional neural networks, Applied Sciences 9 (2019) 2347.
[22] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, D. Brown, Text classification algorithms: A survey, Information 10 (2019) 150.
[23] M. Siino, M. La Cascia, I. Tinnirello, McRock at SemEval-2022 Task 4: Patronizing and Condescending Language Detection using Multi-Channel CNN and DistilBERT, in: Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), Association for Computational Linguistics, 2022.
[24] H. Wu, Y. Liu, J. Wang, Review of text classification methods on deep learning, CMC-Computers, Materials & Continua 63 (2020) 1309–1321.
[25] S. Hashida, K. Tamura, T. Sakai, Classifying tweets using convolutional neural networks with multi-channel distributed representation, IAENG International Journal of Computer Science 46 (2019) 68–75.
[26] S. Aksoy, R. M. Haralick, Feature normalization and likelihood-based similarity measures for image retrieval, Pattern recognition letters 22 (2001) 563–582.
[27] S. Patro, K. K. Sahu, Normalization: A preprocessing stage, arXiv preprint arXiv:1503.06462 (2015).
[28] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017. arXiv:1412.6980.

A. Online Resources

The source code of our model is available via

- GitHub