Character Level Convolutional Neural Network for Indo-Aryan Language Identification

Mohamed Ali
Cairo University, Egypt
mohamedali@aucegypt.edu

Abstract

This paper presents the systems submitted by the safina team to the Indo-Aryan Language Identification (ILI) shared task at the VarDial Evaluation Campaign 2018. The ILI shared task included 5 closely-related languages of Indo-Aryan language family: Hindi (also known as Khari Boli), Braj Bhasha, Awadhi, Bhojpuri and Magahi. The proposed approach is to use character-level convolution neural network to distinguish the four dialects. We submitted three models with the same architecture except for the first layer. The first system uses one-hot char representation as input to the convolution layer. The second system uses an embedding layer before the convolution layer. The third system uses a recurrent layer before the convolution layer. The best results were obtained using the first model achieving 86.27% F1-score, ranked the fourth among eight teams.[1]

1 Introduction

Indo-Aryan is the largest and the dominant language families in the Indian subcontinent. There is no single classification of these languages. Indo-Aryan Language Identification shared task concerned with identifying 5 closely-related languages of Indo-Aryan language family: Hindi (also known as Khari Boli), Braj Bhasha, Awadhi, Bhojpuri and Magahi. This is the first iteration of the ILI shared task at VarDial.

In this paper we present the safina team's submissions for the 2018 ILI shared task which was organized as a part of Vardial Evaluation Campaign 2018 (Zampieri et al., 2018). We have used Char-level Convolutional Neural Network approach to identify Indo-Aryan languages using lexical features. Our team ranked the fourth with F1-weighted score 86.27%.

2 Related Work

Language identification has two flavors: identifying language in spoken text and identifying language in written text. Research in Indo-Aryan Language Identification took place in the two flavors. For the spoken form, Rao et al. (2010) developed Auto Associative Neural Network (AANN) model to identify five prominent dialects of Hindi: Chattisgharhi (spoken in central India), Bengali (Bengali accented Hindi spoken in Eastern region), Marathi (Marathi accented Hindi spoken in Western region), General (Hindi spoken in Northern region) and Telugu (Telugu accented Hindi spoken in Southern region). They examined using spectral and prosodic features to distinguish among those dialects and their best results using a combination of both the spectral and prosodic features. Rao and Koolagudi (2011) explored using Support Vector Machines algorithm which performed poorly compared to AANN. Sinha et al. (2014) used a fully connected Feed Forward Neural Network model four dialects of Hindi: Khariboli,

[1]The code for our submissions is available at: https://github.com/bigoooh/ili

Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects, pages 283–287
Santa Fe, New Mexico, USA, August 20, 2018.

Bhojpuri, Haryanvi and Bagheli. They also used spectral and prosodic features to train their classifier.

For the written form, Indhuja et al. (2014) examined using word and character n-grams to discriminate among five languages used in India (Hindi, Sanskrit, Marathi, Nepali and Bhojpuri). They reported that using word unigrams achieved the best results then character trigrams. However, their findings conflicted with Kumar et al. (2018) who found that using character n-grams significantly out performed using word n-grams in discriminating the same five languages.

3 Methodology and Data

3.1 Character-Level Convolutional Neural Network

Convolutional Neural Networks (CNN) were invented to deal with images and it have achieved excellent results in computer vision (Krizhevsky et al., 2012; Sermanet et al., 2013; Ji et al., 2013). Later, it have been applied in Natural Language Processing (NLP) tasks and outperformed traditional models such as bag of words, n-grams and their TFIDF variants (Zhang et al., 2015). The architecture, shown in Figure 1, describes the character-level CNN model we have used in identifying the Indo-Aryan languages. We formulate the task as a multi-class classification problem. Given text transcript $t^{(i)}$ and the corresponding label $l^{(i)}$, we need to predict l using t. We designed a neural network classifier that takes as input the transcript as one-hot encoded array of characters (padded or truncated from the end to match a predefined maximum length corresponding to the length of the network input layer). The network final output is the probability distribution over the 5 Indo-Aryan languages. The network layers are as follows:

- **Input Layer:** mapping each character to one-hot vector.
- Optional **Embedding or Recurrent Layer :** using embedding or GRU recurrent layer to capture the context of the character (Chung et al., 2014). In the embedding layer case, a list of embedded vectors, one for each character, will fed into the convolutional layer. In the recurrent layer case, using the "return sequences" option will allow it to return one hidden state output vector for each input character. The list of the hidden state output vectors will be fed into the convolutional layer.
- **Convolutional Layer:** contains multiple filter widths and feature maps which is applied to window of characters to produce new features. Each convolution is followed by a Rectified Linear Unit (ReLU) nonlinearity and batch-normalization layers (Glorot et al., 2011; Ioffe and Szegedy, 2015) .
- **Max-Pooling Layer:** apply max-over-time pooling operation over the feature map of each filter and take the maximum value as a feature for this filter (Collobert et al., 2011). The max-pooling operation is followed by a dropout layer to prevent over-fitting (Srivastava et al., 2014).
- **Softmax Layers**: represents the probability distribution over the labels.

Depending on our cross-validation results we used the following parameters for the neural network architecture:

- **Sentence maximum length:** 256 characters
- **Embedding length:**32
- **GRU layer unites:**128
- **Convolution filters sizes:** from 2 to 8
- **Convolution filters feature maps:** 256 feature map for each filter
- **Dropout rate:** 0.2

In our implementation, we used Keras framework with TensorFlow as a backend (Chollet and others, 2015; Abadi et al., 2015).

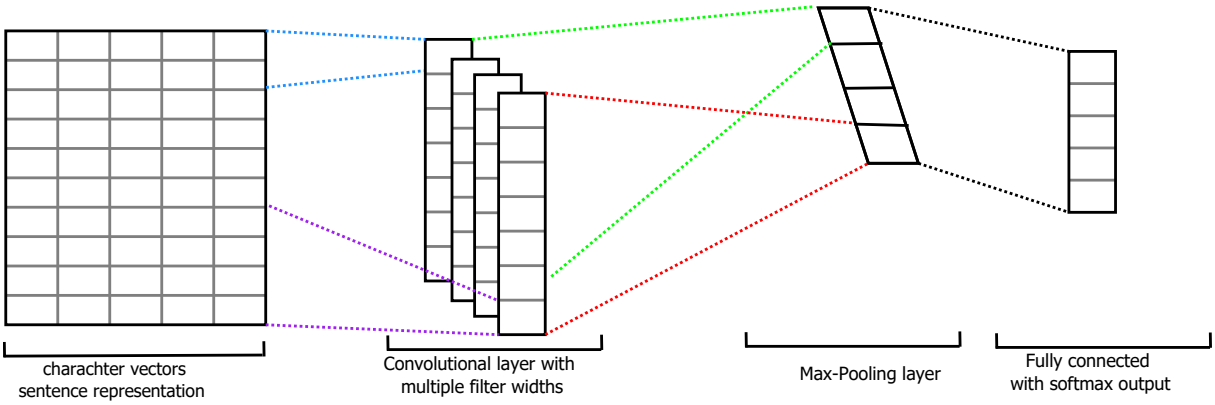

Figure 1: Character-level CNN architecture

3.2 Data

(Kumar et al., 2018) collected a corpus for the five Indo-Aryan languages. The corpus contains about 50,000 sentences distributed over the five languages as shown in Table 1. The authors divided the dataset into: 80% for training and 20% for testing.

Language	number of sentences
Hindi	10,000
Braj	10,000
Bhojpuri	10,000
Magahi	10,000
Awadhi	9,744

Table 1: Number of sentences for each language in the dataset

4 Results

4.1 Cross-Validation Results

We combined the training data and the validation data provided by the shared task to apply 5-fold cross validation. We tested our three different configurations in addition to a TF-IDF features based classifier, Logistic Regression classifier implemented in scikit-learn toolkit (Pedregosa et al., 2011), as a baseline. Results are shown in Table 2.

System	Accuracy
Logistic Regression using TF-IDF features	0.7946
CNN with one-hot encoded input	0.9722
CNN with an embedding layer	0.9718
CNN with a GRU recurrent layer	**0.9774**

Table 2: Cross-validation results

4.2 Test Set Results

Our three runs results are shown in Table 4. We have used the same configuration for three runs except for the input to the convolution layer. In the first run, we fed the one-hot encoded vectors for the sequence of characters directly to the convolution layer. In the second run, we fed the one-hot encoded

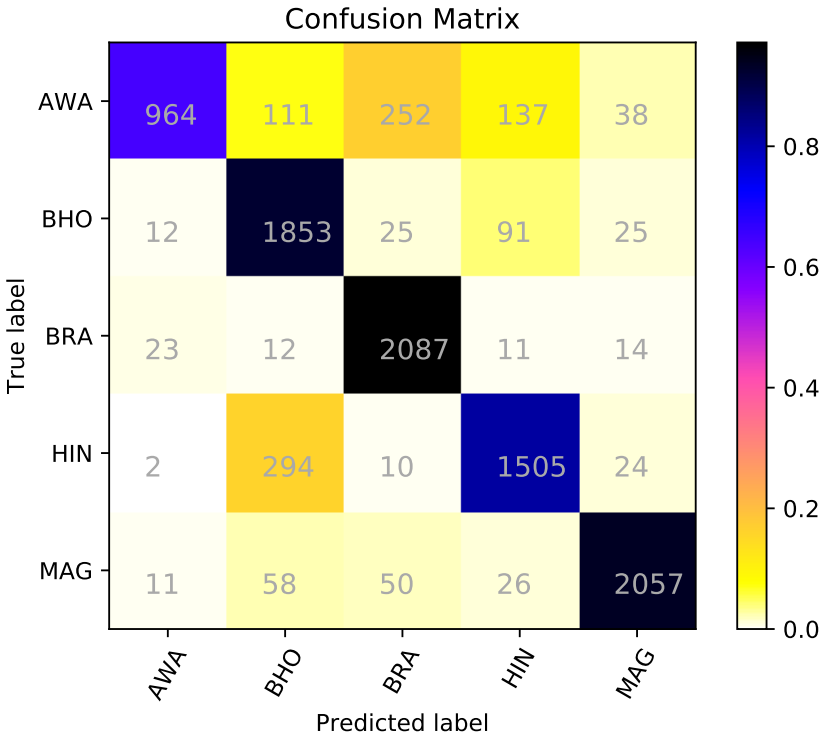

Figure 2: Confusion matrix for our ILI best run

vectors to an embedding layer before the convolution layer. In the third run, we fed the one-hot encoded vectors to a GRU recurrent layer before the convolution layer. As shown in the results, feeding the one-hot encoded representation directly to the convolution layer achieved the best results. This was a surprise for us as it conflicts with our results in the German and Arabic Dialect Identification sub-tasks where using a GRU recurrent layer outperformed feeding one-hot encoded directly to the convolution layer. This may be happened as result of using a large size dataset in the ILI sub-task compared to the ADI and the GDI sub-tasks. In the ILI shared task evaluation, the submitted systems were ranked according to its F1-weighted score. Our team ranked the fourth with F1-weighted score 86.27%. Figure 2 shows the confusion matrix for our best run. From the matrix, we can see that Awadhi and Hindi languages are a little bit confused with the Braj Bhasha and Bhojpuri languages respectively.

System	**F1 (macro)**
Random Baseline	0.2024
CNN with one-hot encoded input	**0.8627**
CNN with an embedding layer	0.8435
CNN with a GRU recurrent layer	0.8260

Table 3: Our three runs results, the best run in bold

5 Conclusion

In this work, we presented our team's three submissions for the ILI shared task. Our approach is to use Character level CNN as a feature extractor from text. Our best submission achieved by feeding the one-hot vector representation of the text as a list of characters directly to the convolution layer without an embedding layer. This abnormal results, compared to our submissions to the the ADI and GDI shared tasks, may be a consequence of using a larger data set in the ILI task.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

François Chollet et al. 2015. Keras. `https://keras.io`.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

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

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323.

K Indhuja, M Indu, C Sreejith, Palakkad Sreekrishnapuram, and PC Reghu Raj. 2014. Text based language identification system for indian languages following devanagiri script. *International Journal of Engineering*, 3(4).

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 2013. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Ritesh Kumar, Bornini Lahiri, Deepak Alok, Atul Kr. Ojha, Mayank Jain, Abdul Basit, and Yogesh Dawar. 2018. Automatic Identification of Closely-related Indian Languages: Resources and Experiments. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

K Sreenivasa Rao and Shashidhar G Koolagudi. 2011. Identification of hindi dialects and emotions using spectral and prosodic features of speech. *IJSCI: International Journal of Systemics, Cybernetics and Informatics*, 9(4):24–33.

K Sreenivasa Rao, Sourav Nandy, and Shashidhar G Koolagudi. 2010. Identification of hindi dialects using speech. *WMSCI-2010*.

Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. 2013. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*.

Shweta Sinha, Aruna Jain, and Shyam S Agrawal. 2014. Speech processing for hindi dialect recognition. In *Advances in Signal Processing and Intelligent Recognition Systems*, pages 161–169. Springer.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Ahmed Ali, Suwon Shon, James Glass, Yves Scherrer, Tanja Samardžić, Nikola Ljubešić, Jörg Tiedemann, Chris van der Lee, Stefan Grondelaers, Nelleke Oostdijk, Antal van den Bosch, Ritesh Kumar, Bornini Lahiri, and Mayank Jain. 2018. Language Identification and Morphosyntactic Tagging: The Second VarDial Evaluation Campaign. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, Santa Fe, USA.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.