ECMAG - Ensemble of CNN and Multi-Head Attention with Bi-GRU for Sentiment Analysis in Code-Mixed Data

Dhanasekaran Prasannakumaran[1], Jappeswaran Balasubramanian Sideshwar[1] and Durairaj Thenmozhi[1]

[1]*Department of Computer Science and Engineering, SSN College of Engineering, Chennai, India*

Abstract

People spend a considerable amount of time on social media platforms consuming information. They share their views and opinions about the subject they consume. The responses could be shared as posts in Facebook and Twitter or through comments on YouTube and the polarity of these posts could be positive or negative or unbiased. The posts or comments in social media are largely present as Romanized English format of multiple languages, commonly referred as code-mixed text. In this work, the authors propose an ensemble framework – **E**nsemble of **C**onvolutional Neural Network and **M**ulti-Head **A**ttention with Bidirectional **G**RU (ECMAG)[1] to map the code-mixed user comments to their corresponding sentiments. The performance of the framework is tested on the Tamil-English Code mixed dataset provided in Dravidian CodeMix FIRE 2021 – Sentiment Analysis for Dravidian Languages in Code-Mixed Text task. The authors use the pre-trained XLM-R model to generate the sub-word embeddings. ECMAG consists of 2 components – Convolutional Neural Network for Texts (CNNT) and Multi-Head Attention pipelined to Bi-GRU (MHGRU). The proposed architecture achieved a F1-score of 0.411.

Keywords

Sentimental Analysis, Code-Mixed text, Transformers, NLP

1. Introduction

The onset of digitization has deemed social media to be a major platform for expressing one's thoughts. Social media platforms like YouTube, Twitter, Facebook, Instagram are used by over 4.4 billion users every day. The amount of information available and accessible is increasing exponentially by the day. Users engage, express and exchange opinions on a subject that interests them. Sentimental analysis aims to identify the polarity of the user's opinion.

With about 122 million daily active users on YouTube consuming more than a billion hours of video content every day, YouTube is one the most widely used social media platform in the world. Users post their views on a video they watched on the comment section. These comments are from a diverse group of people and hence are written in multiple languages. People prefer to

[1]https://github.com/PrasannaKumaran/ECMAG---An-Ensemble-Framework-for-Sentiment-Analysis-in-Code-Mixed-Data/

FIRE 2021: Forum for Information Retrieval Evaluation, December 13-17, 2021, India

prasannakumaran18110@cse.ssn.edu.in (D. Prasannakumaran); sideshwar18151@cse.ssn.edu.in (J. B. Sideshwar); theni1_d@ssn.edu.in (D. Thenmozhi)

use Romanized form of their regional languages to share their thoughts in social media which helps them to easily express their opinions. This results in mixing the vocabulary and syntax of multiple languages in the same sentence which is known as a code-mixed text.

Research studies have been carried out to identify sentiments from monolingual text [1]. Recently, the task of sentimental analysis has extended to code-mixed data and has attracted the research fraternity. In this work, the authors aim to classify the sentiments of YouTube comments in the Tamil-English code-mixed dataset which is part of the 'Dravidian-CodeMix - FIRE 2021 : Sentiment Analysis for Dravidian Languages in Code-Mixed Text' task [2]. The dataset provided consists of code-mixed YouTube comments in Dravidian languages – a family of languages (Tamil, Telugu, Malayalam and Kannada) spoken by 220 million people predominantly in Southern India and Sri Lanka. The vocabulary of these languages are mixed with English to produce the code-mixed text. In this work, the authors propose an ensemble architecture that uses a convolutional neural network and an attention mechanism which is pipelined to a Bidirectional gated recurrent unit layer to classify the comments into one of the given sentiments.

The course of this work is organized as follows. Section 2 elaborates the prominent works in Sentimental analysis of code-mixed data. The details of the dataset used in this work are given in Section 3. The data preprocessing pipeline is presented in Section 4. Section 5 depicts the architecture and elucidates its components. The results of the work are illustrated in Section 6. Finally the authors conclude and discuss the future scope of this work in Section 7.

2. Related Work

Various approaches using Machine Learning (ML) and Deep Learning (DL) have been proposed to solve the task of Sentiment Analysis (SA). Mohammad et al. [3] adopted an ML approach to detect the sentiments of tweets and messages with surface-form, semantic, and sentiment features using a SVM classifier. Giatsoglou et al. [4] proposed a polarity classification model that used hybrid feature vectorization process incorporating lexicon-based features and word embedding based approaches. They employed a SVM classifier with a linear kernel for the classification task.

Designing accurate SA models for multilingual code-mixed text unlike monolingual texts is extremely challenging. Vyas et al. [5] explored different approaches for POS tagging of code-mixed data obtained from Facebook and Twitter. Sharma et al. [6] leveraged various lexicon based approaches for normalization of Hindi-English code-mixed text. A deep learning approach was adopted by Joshi et al. [7], which uses a LSTM to learn sub-word representations to extract the sentiment value of morpheme-like structures. Choudhary et al. [8] proposed a Siamese Network architecture comprising twin Bidirectional LSTM networks that projects the sentences of code-mixed and standard languages to a common sentiment space. Lal et al. [9] proposed a hybrid approach that combines dual encoder RNNs utilizing attention mechanisms, with surface features, yielding a unified representation of code-mixed data for SA. Additionally there has been active research in Offensive language Identification and Hate speech detection on code-mixed social media data [10].

Yadav et al. [11] proposed a zero-shot learning approach that uses cross-lingual and mul-

tilingual embeddings which achieved state-of-the-art scores in Spanish-English code-mixed SA. XML, a state-of-the-art cross-lingual model which learns cross lingual representations in an unsupervised fashion, was proposed by Lample and Conneau [12]. To further improve the performance of XLM, Conneau et al. [13] scaled the size of the model and the data required for pretraining. This resulted in a cross-lingual language model XLM-RoBERTa, a Transformer based masked language model trained on one hundred languages which significantly outperformed Multilingual-BERT(mBERT) [14] and the previous XLM models on a variety of cross-lingual benchmarks. The authors of the papers use the pretrained XLM-RoBERTa model to generate sub-word embeddings for the cross-lingual (Tamil-English) code-mixed data.

3. Dataset

For this work, the authors used the data available in the Dravidian-CodeMix FIRE 2021 [15, 16] database. The data was obtained by crawling Youtube comments. The database contains three different datasets – Tamil-English (Tanglish), Malayalam-English (Manglish) and Kannada-English (Kanglish). Each of the dataset consists of 3 types of code mixed sentences – Inter-Sentential switch, Intra-Sentential switch and Tag switching. The comments are mapped to 5 different labels; Positive, Negative, Mixed Feeling, Unknown state and Unintended language. The authors of this work aim to predict the sentiments of Tamil-English code-mixed text. The summary of the dataset is illustrated in Table 1.

Table 1
Tamil-English Dataset Summary

Sentiment	Train Set	Development Set	Test Set
Positive	20070	2257	2546
Negative	4271	480	477
Unknown State	5628	611	665
Mixed Feelings	4020	438	470
Not-Tamil	1667	176	244

4. Data Preprocessing

The code mixed data provided is extremely noisy. It contains repeated words, emojis, unaccounted words (i.e. words not available in the English dictionary), hashtags, user mentions and obscene words. To handle the inconsistency, the authors propose an extensive data cleaning/preprocessing pipeline to process the raw text.

The authors use Ekphrasis [17] : a collection of lightweight text tools primarily built for processing text data from social medial platforms like Twitter and Facebook. This tool is used for word normalization, word segmentation (for splitting hashtags) and spell corrections. Numbers, hashtags, all caps, extended, repeated and censored words are annotated appropriately.

The text is processed serially and the steps involved in preprocessing is illustrated in Figure 1. Firstly, the sentence is tokenized and the English characters are converted to lower case. The emoji library [18] is used to convert the pictogram (emoji) to words that describe the emotion. Next, the word is checked for its presence in the English dictionary. If found, the word is processed using the Ekphrasis [17] tool. Otherwise, it indicates that the text is either in code-mixed form or in a foreign language. Further, this word is transliterated to its corresponding Dravidian script (Tamil) which is carried out using the google transliteration tool [19]. The sentences that correspond to the unintended language category are not processed in the proposed pipeline.

Hence, a refined text is obtained with either only English words or Tamil words or both. This pipeline therefore mitigates the noise present in code-mixed data. Figure 4 illustrates the text before and after preprocessing.

Figure 1: Pre-processing pipeline

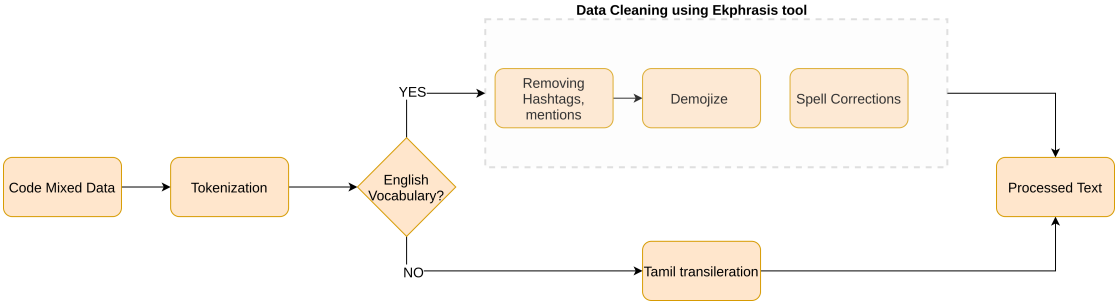


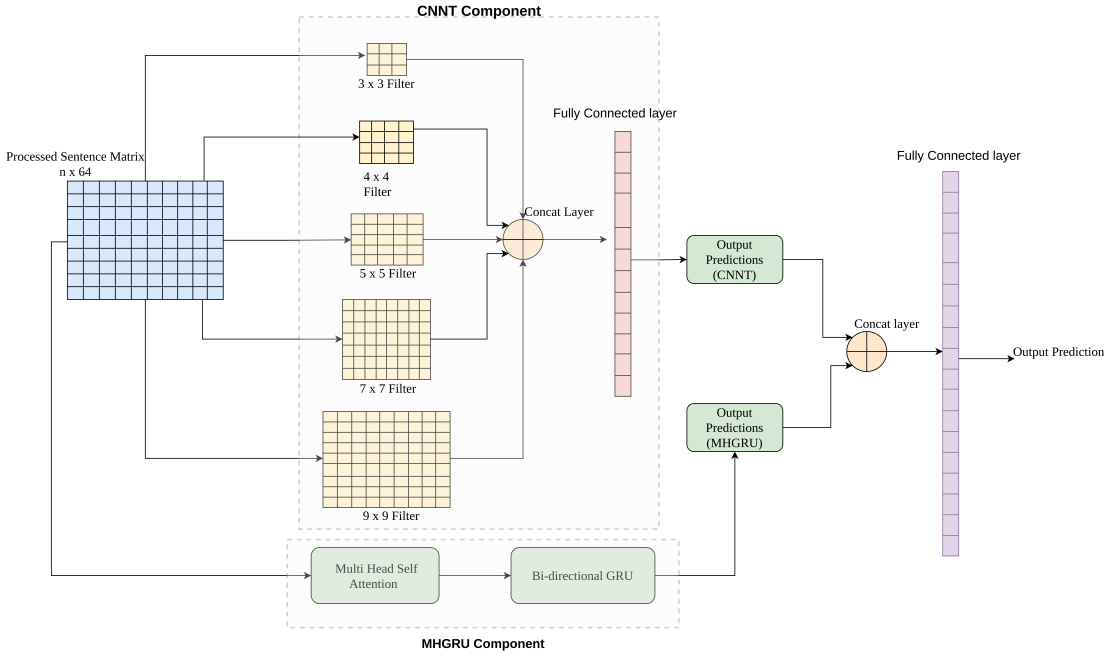
Figure 2: Text preprocessing

Original text	Processed text
Vijay Annaa Ur Maasssssss Therrrrriiiiii	விஜய் அண்ணா ur mass <elongated> தேறி <elongated>
Ithu yethu maathiri illama puthu maathiyaala irukku	இது எது மாதிரி இல்லாம புது மாத்தியால இருக
Yuvan bgm vera level i love u yuvan anna	யுவன் பிகும் வேற level i love u யுவன் அண்ணா
2018 la intha teaser a pathavanga like pannunga	<number> la இந்த teaser a பாத்தவங்க like பண்ணுங்க
you are anti indian tamil 🤣	you are anti இந்தியன் தமிழ் <rolling_on_the_floor_laughing>

5. Architecture

The processed text comprises of other languages's and/or English script. To obtain the word embeddings of multilingual text, the authors used the XLM-RoBERTa (XLM-R) model. XLM-R is a transformer-based masked language model trained on one hundred languages. In this work, xlm-roberta-base model was used. The pre-processed text is tokenized into sub-words using the XLM-R vocabulary. The IDs of these sub-words are then fed to a XLM-R encoder module to obtain the sub-word embeddings which are used as inputs for the proposed architecture.

Figure 3: ECMAG architecture



The authors propose an ensemble framework **ECMAG** (illustrated in Fig 3) which consists of 2 components – Convoloutional Neural Network for Texts (CNNT) and Multi-Head Attention pipelined to Bi-GRU (MHGRU). The details of the components are elucidated in the following sections.

5.1. Convoloutional Neural Network for Texts (CNNT)

The first component is a Convolutional Neural Network (CNN). Several researches [20, 21, 22] have considered using CNN for text classification. CNN was used since it takes into account the ordering of the words and the context in which each word occurs. The sub-word embeddings from XLM-R are passed through a 2D CNN. In this work, the authors considered using 5 filters of different sizes (3, 4, 5, 7, 9). The outputs from the individual 2D CNNs are passed through a max pooling layer. Finally, the outputs from the pooling layers are concatenated and passed through a fully connected layer and the output prediction O_{CNNT} from this component is obtained.

5.2. Multi-Head Bi-GRU (MHGRU)

Attention mechanism can be described as the weighted average of (sequence) elements with weights dynamically computed based on an input query and element's key. Query (Q) corresponds to the sequence for which attention is paid. Key (K) is the vector used to identify the elements that require more attention based on Q. The attention weights are averaged to obtain

the value vector (V). A score function (1) is used to determine the elements which require more attention. The score function takes Q and K as input and outputs the attention weight of the query-key pair. In this work the authors consider using the scaled dot product proposed by Vaswani et al. [23].

$$SelfAttention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

The scaled dot product attention allows the deep learning network to attend over a sequence. However, often there are multiple different aspects to a sequence, and these characteristics cannot be captured by a single weighted average vector. Therefore the authors employed Multi-Head Attention (MHA) [23] which uses multiple different query-key-value triplets (heads) on the same features. Self-Attention (used in this work) first introduced by Luong et al. [24] is an attention mechanism relating different positions of a single sequence in order to compute a representation of the same sequence. Since self-attention was used Q, K and V are initialized with the same sentence (sequence) and the corresponding matrices are transformed into n sub-queries, sub-keys and sub-values and are then passed through the scaled dot product (Equation (1)) attention independently. The attention outputs from each head are then combined and the final weight matrix (W^O)is calculated.

The output from the MHA layer is then pipe-lined through a Bi-directional GRU layer. The output from the Bi-GRU layer is then passed through a fully connected layer and finally through a Softmax layer to generate the predictions. Thus, the output prediction O_{MHGRU} from this component is obtained.

$$\begin{gathered} MultiHead(Q, K, V) = Concat(head_1, \ldots, head_n)W^O \\ \text{where } head_i = SelfAttention(QW_i^Q, KW_i^K, VW_i^V), \\ W^Q, W^K, W^V \text{ are the weight matrices of Q, K and V respectively} \end{gathered} \tag{2}$$

The output predictions from each of the components are concatenated and passed through a fully connected layer to obtain the final prediction as illustrated in Equation (3) .

$$\mathbf{F} : \Delta(O_{CNNT} \oplus O_{MHGRU}) \rightarrow Y \tag{3}$$

6. Results

Experimental Settings : The performance of ECMAG is evaluated based on weighted averaged Precision, weighted averaged Recall and weighted averaged F-Score. The following are the hyper-parameter settings used in ECMAG: maximum sequence length : 64, batch size : 128, CNN output dimension : 5, dropout : 0.3, number of filters : 100, filter sizes : [3, 4, 5, 7, 9], loss function: cross entropy loss, optimizer : Adam, word embedding dimension : 768, GRU hidden size : 32.

Table 2 illustrate the validation results obtained using ECMAG. To validate the importance of the components proposed in the architecture, the results obtained from individual components are also listed in Table 2. The proposed model achieved the following scores on the test data as illustrated in Table 3.

Table 2
Validation Results

Model	Validation						
	Class F1-Score					Weighted average F1-Score	Accuracy %
	Positive	Negative	Mixed Feeling	Unknown State	Not-Tamil		
CNN	0.841	0.287	0.407	0.048	0.021	0.540	59.04
MHGRU	0.827	0.231	0.3894	0.130	0.161	0.534	57.50
ECMAG	0.872	0.267	0.329	0.110	0.0025	0.541	59.57

As the proposed architecture uses word embeddings from a pre-trained XLM-RoBERTa model without fine tuning it to the dataset in hand, the reported scores are only closer to the baseline scores of the task. Fine tuning ECMAG to the given code-mixed dataset would indeed help in capturing the finer meanings and contexts of the sub-words in their embeddings, which in turn would enhance the performance of the model.

Table 3
Test Results

Model	Test		
	Precision	Recall	F1 score
Framework	0.382	0.449	0.411

7. Conclusion

In this work, the authors propose and successfully test an ensemble architecture – ECMAG on the Tamil-English code-mixed dataset to identify the sentiment expressed in YouTube comments. XLM-RoBERTa model was used to obtain the sub-word embedding which was used as inputs to each of the components. ECMAG achieved the following scores: Precision : 0.382, Recall : 0.449 and F1 score : 0.411 on the test data. For future work, the authors aim to process the text further to handle different dialects and slang in Dravidian languages. Fine-tuning the XLM-RoBERTa pre-trained model for the task in hand is another prospective area of work to improve the performance of the model. Additionally the authors aim to tackle the native imbalance present in the dataset between categories. The authors also suggest building an interpretable machine learning model to provide insights on what basis the predictions (sentiments) were made.

References

[1] S. Banerjee, B. Raja Chakravarthi, J. P. McCrae, Comparison of pretrained embeddings to identify hate speech in indian code-mixed text, in: 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020, pp. 21–25. doi:10.1109/ICACCCN51052.2020.9362731.

[2] R. Priyadharshini, B. R. Chakravarthi, S. Thavareesan, D. Chinnappa, T. Durairaj, E. Sherly, Overview of the dravidiancodemix 2021 shared task on sentiment detection in tamil, malayalam, and kannada, in: Forum for Information Retrieval Evaluation, FIRE 2021, Association for Computing Machinery, 2021.

[3] S. M. Mohammad, S. Kiritchenko, X. Zhu, Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets, CoRR abs/1308.6242 (2013). URL: http://arxiv.org/abs/1308.6242. arXiv:1308.6242.

[4] M. Giatsoglou, M. Vozalis, K. Diamantaras, A. Vakali, G. Sarigiannidis, K. Chatzisavvas, Sentiment analysis leveraging emotions and word embeddings, Expert Syst. Appl. 69 (2017) 214–224.

[5] Y. Vyas, S. Gella, J. Sharma, K. Bali, M. Choudhury, Pos tagging of english-hindi code-mixed social media content, in: EMNLP, 2014.

[6] S. Sharma, P. Srinivas, R. C. Balabantaray, Text normalization of code mix and sentiment analysis, in: 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2015, pp. 1468–1473. doi:10.1109/ICACCI.2015.7275819.

[7] A. Prabhu, A. Joshi, M. Shrivastava, V. Varma, Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text (2016).

[8] N. Choudhary, R. Singh, I. Bindlish, M. Shrivastava, Sentiment analysis of code-mixed languages leveraging resource rich languages, CoRR abs/1804.00806 (2018). URL: http://arxiv.org/abs/1804.00806. arXiv:1804.00806.

[9] Y. K. Lal, V. Kumar, M. Dhar, M. Shrivastava, P. Koehn, De-mixing sentiment from code-mixed text, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, Association for Computational Linguistics, Florence, Italy, 2019, pp. 371–377. URL: https://aclanthology.org/P19-2052. doi:10.18653/v1/P19-2052.

[10] K. Yasaswini, K. Puranik, A. Hande, R. Priyadharshini, S. Thavareesan, B. R. Chakravarthi, IIITT@DravidianLangTech-EACL2021: Transfer learning for offensive language detection in Dravidian languages, in: Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages, Association for Computational Linguistics, Kyiv, 2021, pp. 187–194. URL: https://aclanthology.org/2021.dravidianlangtech-1.25.

[11] S. Yadav, T. Chakraborty, Zera-shot sentiment analysis for code-mixed data, Proceedings of the AAAI Conference on Artificial Intelligence 35 (2021) 15941–15942. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17967.

[12] G. Lample, A. Conneau, Cross-lingual language model pretraining, CoRR abs/1901.07291 (2019). URL: http://arxiv.org/abs/1901.07291. arXiv:1901.07291.

[13] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, V. Stoyanov, Unsupervised cross-lingual representation learning at scale, CoRR abs/1911.02116 (2019). URL: http://arxiv.org/abs/1911.02116. arXiv:1911.02116.

[14] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: NAACL, 2019.

[15] B. R. Chakravarthi, R. Priyadharshini, S. Thavareesan, D. Chinnappa, T. Durairaj, E. Sherly, J. P. McCrae, A. Hande, R. Ponnusamy, S. Banerjee, C. Vasantharajan, Findings of the Sentiment Analysis of Dravidian Languages in Code-Mixed Text 2021, in: Working Notes

of FIRE 2021 - Forum for Information Retrieval Evaluation, CEUR, 2021.

[16] B. R. Chakravarthi, R. Priyadharshini, V. Muralidaran, N. Jose, S. Suryawanshi, E. Sherly, J. P. McCrae, Dravidiancodemix: Sentiment analysis and offensive language identification dataset for dravidian languages in code-mixed text, CoRR abs/2106.09460 (2021). URL: https://arxiv.org/abs/2106.09460. arXiv:2106.09460.

[17] C. Baziotis, N. Pelekis, C. Doulkeridis, Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis, in: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), Association for Computational Linguistics, Vancouver, Canada, 2017, pp. 747–754.

[18] T. Kim, Emoji, 2014. URL: https://github.com/carpedm20/emoji/.

[19] G. NC, Googletrans: Free and unlimited google translate api for python. translates totally free of charge., 2020. URL: https://py-googletrans.readthedocs.io/en/latest/.

[20] S. Lai, L. Xu, K. Liu, J. Zhao, Recurrent convolutional neural networks for text classification, in: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, AAAI'15, AAAI Press, 2015, p. 2267–2273.

[21] J. Wang, Z. Wang, D. Zhang, J. Yan, Combining knowledge with deep convolutional neural networks for short text classification, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI'17, AAAI Press, 2017, p. 2915–2921.

[22] S. Moriya, C. Shibata, Transfer learning method for very deep cnn for text classification and methods for its evaluation, in: 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), volume 02, 2018, pp. 153–158. doi:10.1109/COMPSAC.2018.10220.

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, 2017. arXiv:1706.03762.

[24] M.-T. Luong, H. Pham, C. D. Manning, Effective approaches to attention-based neural machine translation, 2015. arXiv:1508.04025.