MITRE at SemEval-2019 Task 5: Transfer Learning for Multilingual Hate Speech Detection

Abigail S. Gertner, John C. Henderson, Amy Marsh,
Elizabeth M. Merkhofer, Ben Wellner and **Guido Zarrella**
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730-1420, USA
{gertner,jhndrsn,amarsh,emerkhofer,wellner,jzarrella}@mitre.org

Abstract

This paper describes MITRE's participation in SemEval-2019 Task 5, *HatEval: Multilingual detection of hate speech against immigrants and women in Twitter*. The techniques explored range from simple bag-of-ngrams classifiers to neural architectures with varied attention mechanisms. We describe several styles of transfer learning from auxiliary tasks, including a novel method for adapting pre-trained BERT models to Twitter data. Logistic regression ties the systems together into an ensemble submitted for evaluation. The resulting system was used to produce predictions for all four HatEval subtasks, achieving the best mean rank of all teams that participated in all four conditions.

1 Introduction

The popularity of social media allows anyone to post their thoughts and opinions for all to see. While the vast majority of these communications are benign, there are those who express hateful or threatening messages online. The identification of hate speech (Fortuna and Nunes, 2018; Schmidt and Wiegand, 2017) on platforms like Twitter is of particular interest for law enforcement and to social media companies who wish to remove accounts with offending content from their sites. Automating the identification of hate speech will allow platforms to flag and remove content much more quickly and effectively.

In this effort we explored neural transfer learning techniques, including word embeddings and fine-tuning of models trained with diverse auxiliary tasks. We built and compared models employing soft attention over sequences and multi-headed self-attention. We also present a novel task to aid in performing additional pre-training of BERT (Devlin et al., 2018) for domain adaptation to Twitter data.

2 Task, Data and Evaluation

HatEval was a shared task organized within SemEval-2019 (Basile et al., 2019). The primary task was detection of hate speech in Twitter, specifically against immigrants and women. This multilingual shared task was organized into two sub-tasks, each presented in both English and Spanish, for a total of four sub-task evaluations.

Task A The first sub-task was simply to identify tweets containing hate speech against immigrants or women. The official metric used for this binary classification task was *macro-averaged F1 score*, in which the F1 scores are calculated for both the positive *hate speech* and negative *not hate speech* classes and then those two scores are averaged.

Task B The second sub-task involved the detection of two specific aspects of hate speech: whether it is targeted at an individual vs. a group of people, and whether it expresses aggression on the part of the author. In this annotation scheme, there is a dependency between these two categories and the hate speech label used in Task A, as tweets could only be labeled as positive for targeting or aggression if they were positive for hate speech. The official metric used for Task B was *Exact Match Ratio* (EMR), which is the proportion of tweets that are labeled correctly for all categories (hate speech, targeting, and aggression). Another way to think of this is as a five-class classification problem where the classes are (H=0, T=0, A=0), (H=1, T=0, A=0), (H=1, T=0, A=1), (H=1, T=1, A=0), (H=1, T=1, A=1). EMR on predicting the three classes separately is equivalent to accuracy on this five-class classification.

Dataset Characteristics The English datasets consisted of 9000 tweets for train, 1000 for dev, and 3000 for test. The Spanish datasets were half the size of the English, with 4500 tweets for train, 500 for dev, and 1500 for test.

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 453–459
Minneapolis, Minnesota, USA, June 6–7, 2019. ©2019 Association for Computational Linguistics

Cursory examination revealed drastic differences between the training and test sets, particularly in English. The pejorative term *bitch* appeared in 12% of the training tweets vs. 48% of the test tweets. The hashtags *#BuildThatWall* or *#BuildTheWall* appeared at rates of 6% and 23% in train and test, respectively. Likewise, *#MAGA* was in over 12% of the test set tweets but in under 3% of the training set messages. Thus the English test set appears to be dominated by a handful of heavily represented phenomena.

Different annotation strategies appear to have been used on the training and test sets as well. While tweets mentioning *#BuildThatWall* or *#BuildTheWall* were annotated as hate speech 98% of the time in the training set, this number is 35% on the test set. Similarly, tweets containing *bitch* were labeled as hate speech 78% of the time in the training set vs. 43% of the time in the test set.

The use of hashtags differs markedly between languages. Hashtags are much more frequent in the English training data than the Spanish training data, with English tweets 2.6 times more likely to contain at least one tag, and with tags occurring in English at 4.1 times the rate in Spanish. In the English training data, the most frequent ten hashtags were 23% of the overall total and tended towards American political topics. In Spanish, the top ten tags account for only 8% of the total, exhibiting a much longer and sparser tail.

3 System Overview

For each task, we created an ensemble of systems, each of which independently predicted the classes. The component systems are described in the following eight sections, after which we describe the procedure for building and testing the ensembles. All component systems described below treated Task B as a five-class prediction problem, and with the exception of two BERT-based systems, were trained to address Task A and Task B simultaneously.

Data and resources SemEval organizers provided training and development sets for English and Spanish. Planning to build ensembles, we shuffled and split out 10% of the training for calibrating models in the ensembles (*calibration set* from here on). Components were trained using the remaining 90% of the training sets provided, with hyperparameter search and validation using the full development sets or via cross-validation. We did not use any additional supervised datasets.

The *BiLSTM*, *Name Embedding*, and *Hashtag Prediction* models incorporated pre-trained `word2vec` (Mikolov et al., 2013) language-specific embeddings that we trained on 1558 billion English and 444 million Spanish tweets collected from 2011 to 2018. In both cases we applied `word2phrase` twice to identify phrases of up to four words, and trained a skip-gram model of size 256, using a context window of 10 words and 15 negative samples per example.

For Task A, all of our component systems and ensembles included a post-processing step to select the best threshold score for classifying hate speech in order to achieve the maximum macro-averaged F1 score on the development set.

3.1 BiLSTM with Attention

We trained several heavily regularized single-layer Bidirectional LSTM (Hochreiter and Schmidhuber, 1997) models to learn a tweet representation with soft attention (Bahdanau et al., 2014) over a sequence of pre-trained token embeddings. Hyperparameter experimentation with Spearmint (Snoek et al., 2012) suggested that a shallow network with attention outperformed deeper, stacked networks and networks without attention. Our attention layer learns to weight context-aware representations of each timestep of the input.

We trained one architecture for the English tasks and two architectures for Spanish, although the second was ablated from our Task A ensemble. The models were identical in structure and differed only in hyperparameters. All models were constructed with spatial dropout over a frozen embedding layer, followed by an embedding transform, one bi-directional LSTM layer with dropout, an attention layer, and a fully-connected hidden layer with dropout.

In each of these models, the NLP representation was used as input to a small prediction network of latent predictions and residual connections described in Section 3.5.

3.2 Name embeddings

This model added a name embedding input to our BiLSTM described above, in an effort to better model the demographics of the individuals addressed within a tweet.

We trained our name vectors using the word2vec objective. Each context was made up of multiple usernames a single Twitter user had employed during a multi-year longitudinal sample of random tweets streamed from the platform. This resulted in a vocabulary of approximately two million name pieces, which includes common names as well as alternate spellings using special characters, symbols, emoji, and other text entered in the *user name* field.

We extracted all substrings of at least length 3 from each username mention in a tweet and included any of them that were in our name embedding vocabulary as input to our model. We applied a learned transformation to each embedding and created a weighted combination with an attention layer. This was concatenated with a hidden representation constructed with the BiLSTM architecture described in Section 3.1. This concatenation was the input to the prediction network described in Section 3.5.

The Spanish name embedding was comprised of dropout over frozen embeddings, a dense embedding transform, and an attention layer. For English, only an attention layer over the frozen embeddings was used. The hyperparameters from our best English model were used in the BiLSTM architecture for both languages.

3.3 DeepMoji

The DeepMoji model developed by Felbo et al. (2017) predicts the emoji removed from an English-language tweet text. The authors train their RNN model on 1274 million tweets for a set of 64 emojis. Using varying degrees of finetuning and newly initialized layers, they test their distantly supervised models on several benchmark datasets for detecting emotion, sentiment, and sarcasm. The model's best results used their *chain-thaw* fine-tuning method, which iteratively unfreezes and trains layers for the new objective. The authors distribute their trained model for the emoji prediction task.

We experimented with both *chain-thaw* training and models that were frozen until the final layer of abstraction in DeepMoji. The pre-trained model has a vocabulary that omits many of the hashtags and usernames that were important for our task. Our best model used 0.75 dropout over the output of a frozen DeepMoji model and three fully connected layers of sizes 512, 256, and 128 before the annotation constraint adapter. *Chain thaw* models performed poorly and were ablated from our Task A submission. DeepMoji models are only included in our English ensembles.

3.4 Hashtag prediction network

Following Zarrella and Marsh (2016), we implemented a recurrent neural network classifier that was pre-trained via an auxiliary masked hashtag prediction task. We extracted 30 of the top hashtags found in the training data, with 15 selected from both the hate speech positive and negative classes. Then we searched for the fifteen nearest neighbors of each tag via cosine similarity in embedding space, using vectors described in Section 3. After removing duplicates, this resulted in 136 English and 132 Spanish hashtags. We downloaded up to 1,000 recent tweets containing each hashtag from Twitter's public search API, resulting in 11,539 English tweets and 12,504 Spanish tweets. Tweets were stripped of the target hashtag(s), and each corpus was divided into a training and development set using a 90/10 split.

The sequence of vector representations of the tokens in each tweet served as the input to a neural network with a 128 LSTM units followed by a dense softmax layer over the possible candidate hashtags. Both the word embeddings and the recurrent layer were tuned. These models correctly predicted development set hashtags with 50.3% accuracy on the English data and 56.6% accuracy on the Spanish data.

The trained weights were extracted from this network and used to initialize the five-way hate speech classifier for Task B, described in Section 2, which additionally saw as input the one-hot representations of the 600 most frequent unigrams and 300 most frequent bigrams in the training data, each followed by a fully-connected dense layer. The size of each fully connected layer and amount of dropout were experimentally determined using Spearmint (Snoek et al., 2012) to maximize performance on the competition metrics on our development set.

3.5 Annotation constraint adapter

Both Task A and Task B had annotation constraints based on latent variables. In Task A, hate speech (H) was not marked as true unless the tweet was directed at women (W) or immigrants (I). In Task B, aggression (A) and individual targeting (T) were not marked as true unless hate speech directed at women or immigrants was present. Even though W and I are not directly represented in our

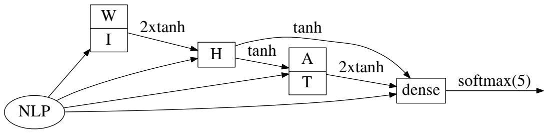

Figure 1: An annotation constraint adapter.

datasets, we believe they are latent variables that can be discovered in the NLP representation. Figure 1 shows an adapter we placed at the end of several systems to encourage the network to learn these constraints. While it doesn't enforce the constraints, it sets up a principled graphical model that encourages the network to learn them. Of course, nothing prevents the network from learning to model other things with this topology. Fair comparisons to stacked dense layers with the same number of parameters showed that the network with this topology performed better.

The upside to the design of a network like this is that the removal of the H switch might yield more general-purpose A and T classifiers.

3.6 Pre-training BERT with Twitter data

Pre-trained language models such as BERT (Devlin et al., 2018) have been demonstrated to achieve state of the art performance on a range of language understanding tasks. BERT uses a transformer encoder model (Vaswani et al., 2017) and pre-trains the model using two complementary objectives: masked language model, and next sentence prediction. The pre-trained model may then be fine-tuned on labeled data (in this case the HatEval dataset) to perform a downstream task.

For English, we used the BERT-Large model, which has 24 layers, 1024 hidden layer size, and 16 self-attention heads. For Spanish, we used the smaller multilingual BERT, with 12 layers, 768 hidden layer size, and 12 self-attention heads. The English BERT is trained on Wikipedia and BooksCorpus (Zhu et al., 2015), while the multilingual model is trained on Wikipedia from multiple languages. As the language in these sources is likely to be quite different from the language commonly used on Twitter, we elected to perform additional pre-training using a corpus of tweets collected during the same time period as the HatEval training dataset (October 2017 - September 2018). All of the pre-training experiments described below started from the TensorFlow model checkpoints downloaded from (Google Research, 2018).

Since the tweets in our collection are not sequential, they cannot be used for the next sentence prediction that BERT uses to learn sentence relationships. We therefore began by running 20k steps of additional pre-training using only the masked language model task.

		none	MLM	descriptions	names
En	A	79.1	**81.2**	79.7	NA
En	B	66.4	**69.0**	67.9	NA
Es	A	80.7	81.9	**83.3**	82.7
Es	B	74.6	75.0	**76.2**	74.4

Table 1: Scores achieved with pre-training schemes. Due to time constraints, the name-based training was only done on Spanish models.

Next, we hypothesized that replacing the next-sentence prediction task with a task involving predicting some attribute of the *author* of the tweet would provide the model with latent information about the nature of tweets that would allow it to discriminate between different classes of tweets more accurately. We performed 20k additional pre-training steps with the *user description* from the author's Twitter profile standing in for the second sequence in the sentence prediction task. In other words, we trained the network to determine whether a given pair of (tweet text, author description text) were sampled from the same tweet. Finally, we pre-trained a BERT model with the *screen name* of the Twitter user as the secondary prediction task.

Table 1 shows the validation scores for our five-class model under our different pre-training schemes: No additional, pre-training on masked LM only, pre-training MLM + Twitter user descriptions, pre-training MLM + Twitter user screen names. Additional pre-training resulted in increased validation scores on all four tasks, and incorporating user descriptions in place of the next sentence prediction task further resulted in increased scores for both Spanish tasks.

3.7 Maximizing ensemble diversity

During development, we noticed some of the neural network models with high capacity had significantly variance in prediction accuracy based on training with different subsets of the training data, hyperparameter settings or just differences in parameter initialization. Such variance would suggest using model bagging (Breiman, 1996) or other form of variance reduction. However, given the relatively long training times for some of the neural network models, especially those based on

$\sigma\lvert w\rvert$	w	feature
0.81	-2.57	*bitch*
0.30	-1.52	*whore*
0.29	-1.12	_bitch_
0.27	-0.97	*women*
0.26	0.53	*URL*
0.23	-1.37	*hoe*
0.23	-0.67	*!*
0.19	-0.91	*her*
0.19	0.72	*immigrant*
0.17	-0.88	*#buildthatwall*
0.17	0.34	//t.co/
0.15	0.74	[*URL,URL*]
0.15	-0.66	#BuildT
0.14	-0.77	*she*
0.14	-0.30	*a*
0.13	-0.61	*woman*
0.13	0.36	*i*
0.12	-0.63	Illegal
0.12	-0.62	*immigrants*
0.12	-0.40	*this*
0.12	-0.39	igrants
0.10	0.63	[*not,all*]
0.10	0.63	[*all,men*]

Table 2: Top LR *word* and character features.

BERT, using ensemble methods such as bagging directly proved too cumbersome as part of the model development workflow. Instead, we employed a form of negative correlation learning (Liu and Yao, 1999) to train a small ensemble of neural network classifiers within a single architecture. A term was added to the fine tuning cross entropy loss function which encouraged diversity among all pairs of classifiers following Opitz et al. (2016).

3.8 Logistic Regression

Logistic regression (LR) systems were developed as a baseline against which the neural approach would be compared. Had annotators used very simple features such as words or phrases to make decisions, they would have been found in the course of LR training. Some of the systems were good enough to include in the final ensembles.

The vocabulary of the LR system was limited to the training set. Many feature sets were explored during model search. The best models preferred feature *sets* rather than *counts* or *term frequencies*. Word n-grams of length 1-3 and character n-grams to length 8 were all considered, along with skip bigrams. The specifics of the best resulting feature sets are in Table 3. Table 2 shows the most important features from an English Task A LR system, sorted by feature *influence*, the product of feature function standard deviation and model weight. The second column is model weight, with negative weights contributing to a (H=1) decision.

In all cases, a bias term was added and Liblinear (Fan et al., 2008) was used to compute the model. L2 regularization was used to encourage generalization. Cross-validation was used to pick the regularization parameters.

3.9 Ensemble

Many systems were created, and final ensembles were constructed by incremental ablations. An initial *all-in* ensemble was created and tested, then it was tested with each component removed. This process was iterated on the best performing ablated sets until gains were no longer observed. Approximately two thousand total ensembles were created through the ablative search. Two systems were ablated in Task A EN, three in Task A ES, one in both Task B conditions. Those systems are not described in this paper.

Ensembles were constructed using logistic regression on either the classifier outputs or the classifier outputs and final probabilities from the model. One oddity to note is that the ensembles using the probabilities performed better for Task A and the ensembles ignoring the probabilities performed better in Task B.

Table 3 shows ensemble compositions for each of the four tested conditions. The first column, labeled *influence*, indicates the influence that the particular component has on the ensemble. It is the number of cases in which that component's contribution *changes* the outcome of the ensemble. It is calculated by zeroing out all LR weights for that particular component and noting the difference. In English, the BERT models had the most influence, while in Spanish, the influence was more evenly distributed across the components.

4 Results

Table 3 shows performance of our component models and ensembles. The *calibration set factored* column shows the performance of the component on our calibration data. This is the *macro averaged F1* score for Task A and *Exact Match Ratio* for Task B. The *calibration set ablated* column shows the performance of the ensemble when that component is removed and the ensemble parameters are re-optimized. Finally there are the scores we calculated after the evaluation period for each of our components using the released reference sets.

The official scores achieved by our ensembles

		calibration set			test	
language	task	influence	factored	ablated	set	component
En	A		86.5		49.6	combo
		369	**84.1**	**84.6**	**58.5**	BERT w/ MLM, 5-class, constraint adapter
		65	78.6	85.7	42.1	BiLSTM+Attn
		59	74.5	85.7	48.0	DeepMoji
		51	82.6	86.2	52.9	BERT w/ descriptions, 1-class
		21	81.3	86.3	47.0	BERT ensemble diversity
		19	78.2	86.1	34.2	BiLSTM with name embeddings
		8	76.4	85.9	48.0	LR, ngrams 1-3, len 7 chargrams, lowercase
		6	75.5	85.8	44.2	Hashtag prediction
		5	77.2	85.9	47.6	LR, ngrams 1-3, len 7 chargrams, lowercase
En	B		77.3		39.9	combo
		435	**74.1**	**75.1**	41.0	BERT w/ MLM
		215	71.7	75.9	37.4	BERT w/ descriptions, constraint adapter
		65	70.6	75.8	33.3	BiLSTM+Attn
		55	67.9	76.6	**43.1**	DeepMoji chain-thaw
		41	58.7	76.0	23.2	Hashtag prediction
		35	69.3	76.4	29.2	BiLSTM with name embeddings
		23	65.6	77.0	38.7	DeepMoji
		23	68.6	76.9	41.0	LR, ngrams 1-2, len 7 chargrams, lowercase
		16	68.2	76.7	41.1	LR, unigrams, len 5 chargrams, lc, skip bigrams
Es	A		87.3		72.9	combo
		90	81.1	**84.9**	**74.3**	BERT w/ names, 5-class
		79	77.9	85.1	73.4	Hashtag prediction
		50	82.1	86.6	73.4	BERT w/ names, 1-class
		45	83.4	85.0	72.0	BiLSTM+Attn
		39	**84.8**	85.3	**74.3**	BERT ensemble diversity
		35	80.7	86.8	73.3	BERT w/ names, 5-class, constraint adapter
		32	79.6	85.0	71.7	LR, ngrams 2-3, len 4 chargrams, lc
		17	82.2	85.0	73.4	BiLSTM with name embeddings
		15	81.4	86.0	73.7	BERT w/ descriptions, 1-class
Es	B		84.7		67.1	combo
		48	78.4	81.8	59.7	BERT ensemble diversity
		42	77.3	80.7	65.3	BERT w/ descriptions
		40	77.8	82.4	63.6	BiLSTM+Attn
		26	75.8	**80.2**	66.8	BERT w/ names, constraint adapter
		21	75.6	83.6	65.6	BERT w/ names
		20	70.0	82.2	59.4	Hashtag predictions
		20	78.4	82.9	67.4	BiLSTM+Attn
		19	76.4	82.4	**68.8**	LR, ngrams 1-3, chargrams 4-7, lc
		15	76.2	82.0	66.7	BERT w/ descriptions, constraint adapter
		11	**79.1**	83.3	65.6	BiLSTM with name embeddings

Table 3: Ensembles and Components

are 49.6% and 72.9% Macro F1 on HatEval Task A English and Spanish, respectively, and 39.9% and 67.1% EMR on Task B English and Spanish. A full reporting of results is present in Basile et al. (2019). A breakdown of test results shows that our system achieves hate speech detection F1 of 63.9 and 72.7 in English and Spanish, respectively, which ranked 2nd (of 68) and 1st (of 39) within Task A. The rankings within Task B were similar, with mean macro F1 of 61.4 and 77.2 in English and Spanish, respectively, ranking 2nd (of 42) and 1st (of 24). Finally, we note that only 22 out of the 74 participants submitted entries in all four subtasks. Of those 22 teams, these results represent the top mean rank across all subtasks.

5 Conclusion

An ensemble of models was used to classify tweets according to whether they contained hate speech, aggression, and targeting of individuals. The novel contributions include using *name embeddings*, substituting twitter author profile prediction for next sentence prediction in BERT pre-training, and augmenting BERT's fine-tuning loss function with a diversity term to create an ensemble.

There is a discrepancy between the official test set results and our held-out calibration set, particularly in the English subtasks, which we attribute to dataset divergences like those called out in Section 2.

Approved for Public Release; Distribution Unlimited. Public Release Case Number 19-0700

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations Workshop*.

Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Rangel, Paolo Rosso, and Manuela Sanguinetti. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019)*.

Leo Breiman. 1996. Bagging predictors. *Machine learning*, 24(2).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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*, 9:1871–1874.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Paula Fortuna and Sérgio Nunes. 2018. A survey on automatic detection of hate speech in text. *ACM Comput. Surv.*, 51(4):85:1–85:30.

Google Research. 2018. https://github.com/google-research/bert.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yong Liu and Xin Yao. 1999. Ensemble learning via negative correlation. *Neural networks*, 12(10):1399–1404.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*.

Michael Opitz, Horst Possegger, and Horst Bischof. 2016. Efficient model averaging for deep neural networks. In *Asian Conference on Computer Vision*.

Anna Schmidt and Michael Wiegand. 2017. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Guido Zarrella and Amy Marsh. 2016. MITRE at SemEval-2016 task 6: Transfer learning for stance detection. In *SemEval@NAACL-HLT*.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *2015 IEEE International Conference on Computer Vision (ICCV)*.