Judging a Book By Its Back Cover: Book Genre Prediction Using Neural Networks

Suraj Subraveti
University of Massachusetts Amherst
ssubraveti@cs.umass.edu

Abhiram Eswaran
University of Massachusetts Amherst
aeswaran@cs.umass.edu

Abstract

The goal of our project is to predict the genre(s) of books using their summaries. We plan to use two different approaches for doing this: As a baseline, we use a standard multi-layer feed-forward neural network, that takes documents represented as probability distributions over topics using Latent Dirichlet Allocation[1], as input. The second approach is to use skip-thought vectors[9] with feed forward neural nets. The main idea is to use representations of sentences using pre-trained skip-thought vectors, and using these representations to classify book summaries using a feed forward neural network.

1. Introduction

People who buy books on a regular basis often base their decision on book summaries that are at the back. The main purpose of doing this is to get an idea about the genre of the book. We were curious to see if we can train a neural network to do the same. This could be really useful for e-commerce websites to categorize books, and maybe also use this classification to recommend books to users, based on what genre of books they might be interested in.

In this work we propose a feed forward neural network which takes the skip-thought vector representation of book summaries and outputs the possible genres of the box. This model draws global dependencies from the sentences of a book summary to give a skip thought vector. This representation with a feed forward neural network achieves higher accuracy with lesser epochs compared to the baseline model using LDA and feed forward neural nets.

2. Related Work

Iwana et al.[5] have used images of book covers to try and predict genres of books. They use transfer learning to adapt pretrained AlexNet[10] and LeNet[13] model for book genre classification. We believe the summaries of a book give a better representation than covers of a book. We explore the same with this project.

Representing summaries is crucial for our project and we found that Zhou et. al.[20] have used a combination of CNNs and LSTMs to represent sentences. In their network, a simple CNN is used to extract a sequence of higher level phrase representations, which are fed into an LSTM network to obtain sentence representations.This is a good idea because you get the best of both worlds: CNNs are able to learn local response from temporal or spatial data, but lack the ability of learning sequential correlations. On the other hand, LSTM networks are specialized for sequential modeling, but are unable to extract features in a parallel way.

It was also important that the pre-trained word vectors we use are universal feature extractors. Yoon Kim[8] have used a simple convolutional neural network(one single dimensional convolution followed by max pooling over time as in Collobert et. al.[2], followed by fully connected neural networks for classification) over sequences over sequences of pretrained word vectors trained by Mikolov et. al.[14], and found that this simple architecture with little hyperparameter tuning produces state of the art results on many NLP benchmarks. This indicates that pre-trained word vectors are universal feature extractors, that can be used in many classification task. Fine tuning these vectors for specific tasks further improves results.

3. Exploratory Analysis

Having read and come across various successful projects, we knew it was crucial to spend a good amount of time pre-processing our data. With this in mind, we first did some exploratory analysis on the dataset. This helped us get a better understanding of the dataset, and the kind of pre-processing we would need to do.

3.1. Labels

We found that the labels for each example in the dataset were not uniform. Most of the training examples had multiple genres. To be able to use the softmax loss to train our networks, we therefore decided to just pick one of the mul-

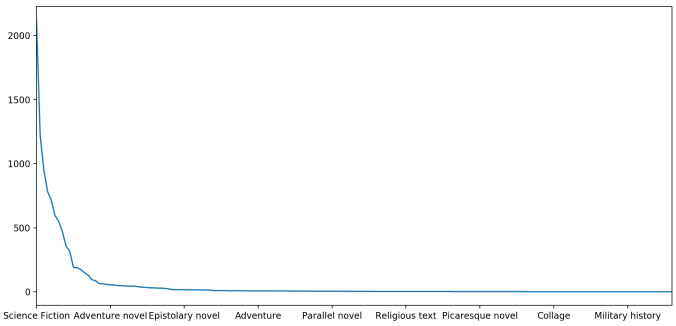

Figure 1. Distribution of genres. The labels on the x axis do not depict all the genres

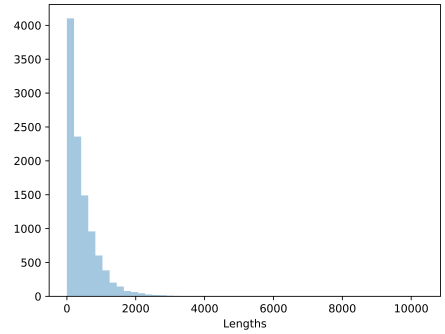

Figure 2. Histogram showing lengths of all the summaries in the dataset

Measure	Value
Mean	456.30
Max	10335
Min	2
Median	295
Mode	61
Standard Dev.	505.89

Table 1. Statistics on the lengths of summaries

tiple listed genres. We picked the first label of all the labels. To understand how these are distributed, we visualized the distribution of examples using a histogram. The results are shown in Figure 1. We found a total of 174 genres, and there was a lot of skew(more than 2000 examples had been tagged Fiction, and many genres had just 1 or 2 examples in the dataset).

This is not clearly not ideal, because either the training data or the test data would contain many more examples of a particular genre, and in each case, we wouldn't be able to determine how good/bad our model is.

We thus decided to leverage the information provided by the multiple labels by changing our loss function from the traditionally used softmax cross entropy to using the *Kullback-Leibler Divergence*[11]. More about this in the next section.

We also found quite a few examples(around 4000 of them) with no labels. We discarded them for simplicity. Initially we thought we could still use them to train our SkipThoughts encoder, but we decided to use a pre-trained model instead due to the training time complexity.

3.2. Lengths of sequences

We were then curious about the number of tokens in each example(book summary). We did some analysis in this regard, and plotted the results. These are depicted in the histogram in Figure 2 and related statistics in Table 1. Table 1 shows some statistics. As you can see from the table, the longest sequence is too long, and the shortest sequence is too short. To make things simpler, we considered summaries that have at least 50 tokens, and at most 2000 tokens.

3.3. Preprocessing

Given the above findings from exploratory analysis, we made certain changes to the data accordingly.

- **Summary lengths** : All data samples whose length were lesser than 50 and greater than 2000 were dropped. Post this processing we were left with 10,500 samples. We split this in 60:20:20 ratio for train, validation and test sets respectively.

- **Genres** : We found that the dataset mentioned a total of 221 genres. Referring to Figure 1, we dropped all genres whose number of occurrences were less than 100. This left us with 23 genres.

- **One-hot encoded genres**: Since we decided to use KL Divergence as our loss, we required a binary representation of each genre to get the true probability distribution. We achieved this by using Sklearn's [17] MultiLabelBinarizer.

4. Approach

4.1. Loss Functions

4.1.1 KL Divergence

Kullback-Leibler Divergence[11]. Kullback-Leibler(KL) divergence is a measure of how one probability distribution diverges from an expected probability distribution. Mathematically,

$$D_{KL}(P||Q) = -\sum_i P(i) \log\left(\frac{Q(i)}{P(i)}\right) \quad (1)$$

if P and Q are discrete probability distributions. The expected probability distribution(P) in this case is the ground truth probability distribution we construct by distributing the probability equally between all the correct labels for a particular example. In a nutshell, we minimize the KL divergence between the output of the network(Q in the formula above)(which is a probability distribution) in each case and the ground truth probability distribution.

4.1.2 Binary Cross Entropy

We decided to use another loss function on the same lines of the standard cross entropy loss(also referred to as categorical cross entropy loss) used in classification problems and tweaked it to work in a multi-class setting. The standard cross entropy loss function is given by:

$$L_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right) \tag{2}$$

where f is the vector of scores for each class, and y_i is the correct class, and f_x indicates the score for class x in the scores vector, and N is the number of examples in the minibatch.

To make this work in our setting, where we predict multiple classes for each example, we use the following loss function:

$$L_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{C}\sum_{j=1}^{C}[y_{ij}\log\left(\hat{y}_{ij}\right)+ \tag{3}$$

$$(1-y_{ij})\log\left(1-\hat{y}_{ij}\right)) \tag{4}$$

where $\hat{y}_{ij}$ is the predicted score for class j of example i, y_{ij} is the actual score(0 for the wrong class, and $1/c$ if example i has c genres listed in the ground truth), N is the number of examples in the minibatch, and C is the total number of classes. When we use binary cross entropy, we try to maximize the correct scores of *all* the correct classes, rather than of one particular class, as with categorical cross entropy

4.2. Models

In this section, we describe the four approaches used in our project. The first is the baseline model which is LDA + feed forward neural net and the others being the average, average-sampled and maxed out Skip-thought vectors with feed forward neural nets.

4.2.1 Latent Dirichlet Allocation

Topic models have been widely used for document summarization. LDA is a generative probabilistic topic model that assumes that documents are generated using a generative process, and given a corpus of data, it tries to infer this process. So, using a topic model, it is possible to represent a document as a probability distribution over a fixed number of hidden/latent topics. Each topic in turn, is a probability distribution over words. This vectorized representation of each document can later be used as input to a feed-forward neural network that acts as a multi-class classifier.

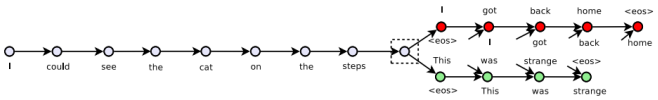

Figure 3. Diagram showing the encoder-decoder architecture

4.2.2 Skip-Thought Embeddings

We encode the book summaries using skip thought vectors[9]. This model uses an encoder-decoder architecture to generate embeddings for sentences. Given a corpus of text, this model uses a triplets of sentences(say (s_{t-1}, s_t, s_{t+1})), to train the encoder and the decoder. s_t is fed into a recurrent neural network (as represented in 3), and the representation generated at the last time step of this RNN is fed into a couple of decoder RNNs, and the entire network is trained to maximize the probability of each of the decoders generating s_{t-1} and s_{t+1}. After training this encoder-decoder, the decoder is discarded, and for any sentence, the representation generated by the encoder is used as the embedding for the sentence. We to use chose a pre-trained encoder trained by the authors on the BookCorpus[21] dataset for two weeks over training one from scratch due to the enormous compute power required to train skip-thought vectors. Sentences that share semantic and syntactic properties are thus mapped to similar vector representations. We used these representations in three different ways.

- **Average vectors**: We take the average of the representations over the entire book summary and fed it as an input to a neural net.
- **Average of sampled sentence vectors**: We randomly sampled 10 sentences from each book summary and averaged it. This was then fed into a feed-forward neural network.
- **Max of sampled sentence vectors**: We randomly sampled 10 sentences from each book summary and took the max of over each feature of these 10 skip-thought vectors. This representation was then fed it into a feed-forward neural network.

4.2.3 The Common Denominator: Feed-forward neural network

We used a standard multi-layer feed forward neural network. Our network has six hidden layers, each with 256 neurons. This network tends to overfit easily, so to prevent that we use L2 regularization[16] on the weights of each hidden layer, and use dropout[18]. In addition to this, we also have a batch normalization[4] layer after the affine transformation in each layer. We initially used ReLU[15] activations for each layer, but observed that the training loss wasn't going down too much. On further investigation, we found that ReLU was thresholding the activations of some of the neurons to 0, and the gradients were 0 for these neurons. We then decided to use a Parametric ReLU[3] activations, with L2 regularization on the parameters to prevent overfitting.

4.3. Evaluation

4.3.1 Jaccard Similarity

A book can have multiple genres. Therefore we used Jaccard Similarity[7] to evaluate our results. The Jaccard similarity coefficient, is the size of the intersection divided by the size of the union of true label and the predicted label sets. Given the two label sets, A and B, the Jaccard similarity coefficient can be expressed mathematically as:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

This is used to compare the set of predicted labels for a sample to the corresponding set of true labels.

4.3.2 Accuracy

- Best 1 Accuracy: Fraction of samples in which at least one of the genres was predicted correctly. We chose to leverage this, because of the number of narrow genres present in the data set.

$$B1A = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left(|y_i \cap \hat{y}_i| \geq 1\right) \tag{5}$$

  where $\mathbb{1}$ is the indicator function, which is 1 if the condition inside is true, and 0 otherwise

- Average Precision: Average precision is the proportion of correctly predicted labels, to the total number of actual labels, averaged over all training examples.

$$AP = \frac{1}{N}\sum_{i=1}^{N} \left(\frac{|y_i \cap \hat{y}_i|}{|\hat{y}_i|}\right) \tag{6}$$

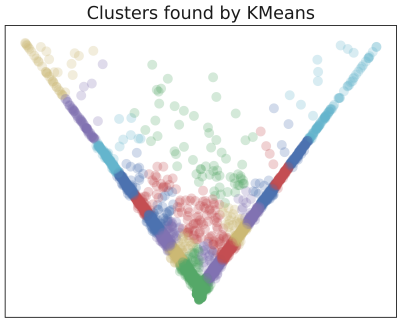

Figure 4. Scatter plot showing the 2D representation of LDA vectors

5. Experiments

5.1. Baseline: LDA with Feed-forward Neural Networks

5.1.1 Clustering LDA vectors

We started by running the book-summaries into Sklearn [17]'s LDA to vectorize the summaries into 200 topics. We then projected these document vectors on to a two dimensional space. In Figure 4, each color represents a different cluster. From the figure, we can conclude that semantically similar documents(i.e. documents belonging to the same cluster) do not end up near each other in a vector space.

5.1.2 KL Divergence vs Binary Cross Entropy

Choosing a good loss function is crucial to get the best results. To compare the performance of KL Divergence and Binary Cross Entropy we plotted their metrics for 6 layer neural network mentioned above. The results for this can be seen in Figure 5 and 6.

From Figure 5 it can be seen, that using KL-Divergence as a loss function leads to a lot of noise and variance in the results. Binary Cross Entropy is more stable and values drop slowly as we overfit. Figure 6 shows the training loss for KL Divergence and Binary Cross Entropy. It can be seen that the loss drops to zero with KL Divergence very fast, while the drop in loss for Binary Cross Entropy is comparatively more stable.

5.1.3 Early stopping

Early stopping is crucial to prevent the network form overfitting. To ensure this we plotted the validation vs training loss curve presented in Figure 7. The figure shows that the network is clearly over fitting with 400 epochs and works best when we stop training at 150 epochs. This is reflective

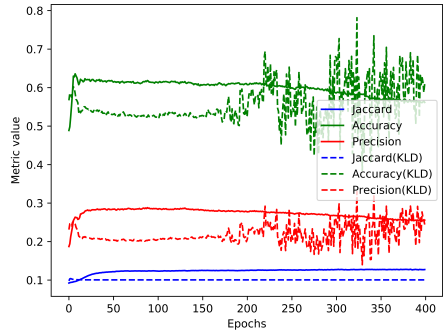

Figure 5. Metrics on the validation set(LDA+FC net) with KLD and BCE

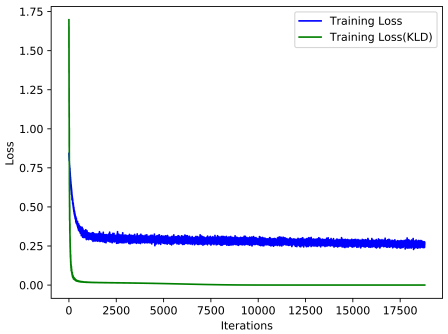

Figure 6. Comparison of KLD and BCE losses(LDA+FC net)

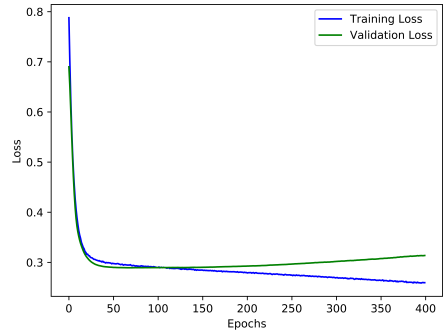

Figure 7. Training v/s validation loss(LDA+FC net)

in Figure 5 where we see the validation accuracy slowly begin to drop after 150 epochs.

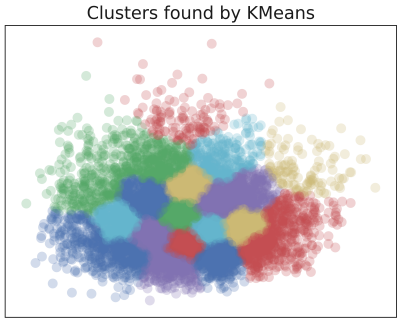

Figure 8. Scatter plot showing the 2D averaged Skip-Thought embeddings

5.2. Skip-thought Embeddings with Feed Forward Neural Network

5.2.1 Clustering Skip-thought vectors

Pre-trained Skip-thoughts were used to represent the book summaries where each summary was a 4800 long vector. We then projected these document vectors on to a two dimensional space. In Figure 8, each color represents a different cluster. From the figure, we can conclude that semantically similar documents(i.e. documents belonging to the same cluster) end up near each other in a vector space indicating that averaging skip thought embeddings of sentences in summaries lead to a better representation for documents than LDA.

5.2.2 Average vectors vs Sampled average vectors vs Sampled maxed out vectors

- **Average over all sentences** The first experiment we tried using skip thought embeddings was to represent a document as the average over skip thought embeddings of all the sentences in each summary. We chose to aggregate sentence vectors in this manner because it has been shown that a simple average works much better than other complex schemes, for NLP tasks like question answering[6]. But crude averaging over all sentences in a summary, penalizes the word representations heavily and most features end up having the same value. This is represented in Figure 9.

- **Average over sampled sentences** We used skip thought embedding of each book summary and randomly sampled 10 vectors from the representation of each book summary. We then averaged over these 10 sentence vectors and fed these to a feed forward network. It was observed that these performed better than simple averaging over all sentences since it gives

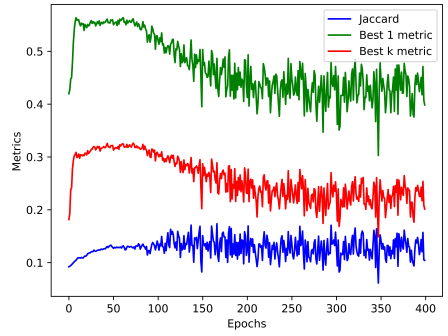

Figure 9. Metrics on the validation set for SkipThoughts averages vectors + FC net

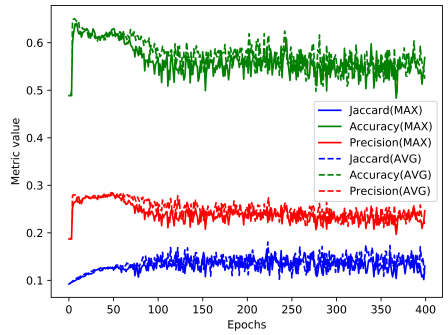

Figure 10. Metrics on the validation set for SkipThoughts averages vectors + FC net

Model	B1A	Precision
LDA + FC Net	0.63475	**0.28716**
ST(Sampled AVG) + FC Net	**0.67112**	0.28609
ST(Sampled MAX) + FC Net	0.63529	0.27914

Table 2. Results on the test set

a more balanced representation of the book summary. This can be seen from the dotted lines in Figure 10.

- **Max over sampled sentences** As an alternative method, we used the same approach as mentioned above and maxed over every feature instead of averaging them. It was observed that this representation performed slightly worse than sampled averaging method mentioned above. This can be seen from the solid lines in Figure 10.

6. Conclusion

After training our model on the training set, and tuning hyperparameters on the validation set, we tested the performance of our trained models on the test set. Table 2 summarizes the results of our trained models on the test set. Below we can also look at a test sample.

6.1. Test Sample

Input:

```
Following on almost immediately from
the events of Seduced by Moonlight, A
Stroke of Midnight begins with Merry
and the Ravens attending a press
conference in the sithen.  This is
highly unusual as the home of the
sidhe is usually off-limits to the
human press.  However, it is felt that
it is more secure than holding the
conference elsewhere.  This opinion
is challenged almost immediately by the
deaths of Beatrice, one of the lesser
fae, and a human reporter.  Merry,
assigned to solve the murders by her
aunt, Queen Andais, opts to bring
in human forensics in the hope that
science might be able to succeed where
magic has so far failed - and bring a
murderer to justice.  The entire novel
takes place within approximately one
day.
```

Predicted Genres:
Science Fiction, Fantasy, Suspense

6.2. What do the results say?

From the table, as expected, the model with documents represented as average of sampled SkipThought embeddings of sentences gets the best accuracy. The performance of both the LDA model and the Skip-Thoughts max model are quite similar. We believe that the performance of the Skip-Thoughts max model is not as good as the Skip-Thoughts average model because in the former case, we are basically trying to predict the genres of each book using only one sentence from the summary. On the other hand, when we average Skip-Thought embeddings for multiple sentences, we encode more information in the document representation, leading to better predictions. Skip-thoughts also performs better than LDA because it uses locality of previous and next sentences as part of its training, to make it a more efficient representation of a sentence than LDA.

6.3. Future Work

Shortage of resources handicapped us from trying other complex models. If we had more compute resources, we would have wanted to try the following ideas:

- The baseline model with LDA gives interesting results and training LDA with higher number of topics

is something we would like to explore. This would help us understand how well the model will perform if we were to represent LDA vectors as a vector of 4800 topics like skip-thoughts.

- Training a Skip-Thoughts encoder for sentences on our corpus, by also leveraging many examples we dropped due to reasons like absence of labels, or examples with very infrequent labels. We suspect that this will yield better results than using an encoder trained on a different dataset(although in this case, it was trained on a similar dataset of novels)
- Explore alternative approaches to representing documents like doc2vec[12]
- Use an LSTM over sequences of Skip-Thought embeddings and Attention Models[19] to predict genres of the book.

References

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, Mar. 2003.

[2] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.

[4] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[5] B. K. Iwana and S. Uchida. Judging a book by its cover. *CoRR*, abs/1610.09204, 2016.

[6] M. Iyyer, J. Boyd-Graber, L. Claudino, R. Socher, and H. Daumé III. A neural network for factoid question answering over paragraphs. In *Empirical Methods in Natural Language Processing*, 2014.

[7] P. Jaccard. Etude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin de la Socit Vaudoise des Sciences Naturelles*, 37:547–579, 1901.

[8] Y. K. Kim. Convolutional neural networks for sentence classification. In *EMNLP*, 2014.

[9] R. Kiros, Y. Zhu, R. Salakhutdinov, R. S. Zemel, A. Torralba, R. Urtasun, and S. Fidler. Skip-thought vectors. *CoRR*, abs/1506.06726, 2015.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60:84–90, 2012.

[11] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann.Math. Statistics*, 22(1):79–86, 1951.

[12] Q. V. Le and T. Mikolov. Distributed representations of sentences and documents. *CoRR*, abs/1405.4053, 2014.

[13] Y. LeCun. Gradient-based learning applied to document recognition. 1998.

[14] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

[15] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 807–814, USA, 2010. Omnipress.

[16] A. Y. Ng. Feature selection, l1 vs. l2 regularization, and rotational invariance. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 78–, New York, NY, USA, 2004. ACM.

[17] 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. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, Jan. 2014.

[19] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy. Hierarchical attention networks for document classification. In *HLT-NAACL*, 2016.

[20] C. Zhou, C. Sun, Z. Liu, and F. C. M. Lau. A c-lstm neural network for text classification. *CoRR*, abs/1511.08630, 2015.

[21] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *arXiv preprint arXiv:1506.06724*, 2015.