Tally and Christina's Awesome Project!!!!!

Tally Portnoi
6.864
tportnoi@mit.edu

Christina Sun
6.864
sunc@mit.edu

Abstract

Currently, question answering forums are commonly used to quickly obtain specific information pertaining to a particular domain. We study approaches for finding similar questions, with the hope that existing answers to questions asked previously can be reused. Automating this process can reduce response times for the user and eliminate repeated work. We use convolutional and recurrent neural networks to encode questions into vector representations, which are then compared using a similarity score. Our models achieve scores comparable to those reported by (Lei et al., 2015). Since annotated data is scarce in most domains, we also explore transfer learning methods for training models using annotated data in the source domain to make predictions in the similar but different target domain. Our adversarial domain transfer model achieves a 12% increase in the AUC score over the direct transfer baseline without any domain adaptation augmentations.

1 Introduction

This project considers the problem of Question Answering (QA) for question answering forums via Question Retrieval and Transfer Learning. Instead of tackling the task of natural language generation, our model searches for similar questions in order to reuse their answers. This behavior mimics human users, who often provide links to similar questions as answers. This provides the question-asker with the information they are looking for and reduces the number of unnecessary, duplicate answers across the forum. For this task, we encode questions consisting of a title and a body into semantic vector representations and compare them via cosine similarity. We also explore adversarial domain adaptation techniques in the case where data pertaining to similar questions is available in one domain, but not another. More specifically, we train the model to learn features that are both invariant between the two domains and useful for the main information retrieval task.

2 Related Work

Research in question retrieval is growing in popularity. For the specific task of question retrieval in question answering forums, Lei et. al. (2016) propose RCNN, a convolutional model with adaptive gated decay and pre-trained using the entire corpus to map questions into semantic vector representations. RCNN is compared to various other encoders such as CNN, LSTM, and GRU as well as baseline models TF-IDF, BM25, and SVM. Ganin et. al. (2015) introduce a simple but effective method for domain transfer by augmenting a model with a gradient reversal layer, and apply it to image classification tasks. Zhang et. al. (2017) perform aspect transfer, where both the source and the target classifiers operate over the same domain by jointly optimizing the objective, combining word reconstruction, relevance labels, transformation layer regularization, source class labels, and domain adversary losses.

3 Question Retrieval Setup

The system is given a corpus of questions from the Stack Exchange AskUbuntu Dataset (Lei et al., 2015) $Q = \{q_1, , q_n\}$ and a training set of questions where each is accompanied with a list of similar questions which are marked by users and a list of random questions which are used as negative training examples. Given a new question q_i consisting of a title and a body, our goal is to retrieve all relevant questions from a candidate set of related questions $Q(q_i)$ which are retrieved using a standard IR engine. The goal of our model is to rank the candidates according to a scoring function that measures the similarity between two questions, such that questions similar to q are ordered first. To accomplish this, we encode the questions into a vector representation using a neural network model and use the cosine similarity between the two representations as the scoring function. We optimize the scoring function by considering annotated data where each sample consists of question, q_i, a question similar to q_i, p_i^+ and a set of questions dissimilar to q_i, Q_i^-. We tune the parameters of the neural network by minimizing the encoder's max-margin loss $\mathcal{L}^{enc}$.

4 Transfer Learning Setup

We transfer the model to perform Question Retrieval on the Stack Exchange Android dataset (Guo, 2017). In addition to the data described above, the system is given a set of questions from the Android corpus for training a discriminator to distinguish between questions from the two domains. More specifically, in addition to the encoder, the model contains a discriminator that classifies each sample as from the Android domain or AskUbuntu domain. Each data sample for the discriminator consists of 20 questions chosen randomly from the Android corpus, labeled with the domain tag 0, and 20 questions chosen randomly from the AskUbuntu corpus, labeled with the domain tag 1. There are two optimizers, one for the encoder as before, and one for the domain classifier. We set a negative learning rate for the domain classifier. Conceptually, if the discriminator is unable to discriminate between Android questions and AskUbuntu questions, the representations learned are generalized enough so that the model does not rank similar questions according to features that are specific to a single domain. The features that emerge are discriminative for the question retrieval task on the source domain, yet invariant when transferring to the target domain. The loss function is $\mathcal{L}^{enc} - \lambda * \mathcal{L}^{dc}$, where λ is a constant and $\mathcal{L}^{dc}$ is the binary cross entropy loss for the domain classifier.

5 Question Retrieval Models

The encoder takes as input each successive token in the question title or body. This process transforms the entire input sequence into a state sequence which is aggregated into the final vector representation via mean pooling. We encode both the question title and body and average the two resulting vectors to obtain the questions final representation.

CNN Convolutional neural networks (LeCun et al., 1998) are commonly used in NLP problems. Filter matrices $W_1, ...W_n$ are convolved with each window of n consecutive words, producing feature maps, which are the output state vectors h_t. We use mean pooling to combine the state vectors into a meaningful representation for the entire sequence.

LSTM Long Short Term Memory cells (Hochreiter and Schmidhuber, 1997) have been applied to various NLP tasks since they are effective in capturing dependencies earlier on in the sequence compared to other models such as vanilla RNN. With each input token, LSTM cells use input, forget, and output gates to choose which information to read and discard from internal memory states.

6 Transfer Learning Models

TF-IDF We compare our adversarial Domain Transfer model to two baselines. The first is the unsupervised TF-IDF model. Android questions (title and body) are converted into vector representations using a tf-idf weighted bag-of-words model. These representations are compared according to cosine similarity.

Direct Transfer The second baseline is direct transfer using models trained on the AskUbuntu dataset and evaluated on the Android dataset, without performing any domain adaptation.

Adversarial Domain Transfer We use the LSTM model described above as the encoder, and a simple feed forward network with three linear layers as the domain classifier. Data samples are fed into the label predictor which encodes the samples and produces cosine similarities, as described previously. The adversary takes as input 20 random samples from the Android corpus and 20 random samples from the AskUbuntu corpus. It encodes the samples using the encoder and feeds the resulting representations into the domain classifier, producing a set of predicted domain labels.

Method	h	$\|\theta_{enc}\|$	n	lr	d	m	$\|\theta_{dc}\|$	lr_2	λ	h_2	h_3
CNN	670	403K	3	1e-3	0	0.5	-	-	-	-	-
LSTM	240	309K	-	3e-4	0	0.5	-	-	-	-	-
Direct Transfer	110	157K	-	3e-4	0.4	0.2	-	-	-	-	-
Adversarial Domain Transfer	110	157K	-	3e-4	0.4	0.2	79K	1e-5	1e-2	300	150
Exploration	-	-	-	-	-	-	-	-	-	-	-

Table 1: Configuration of neural models. h is the hidden dimension of the encoder, $|\theta_{enc}|$ is the number of parameters for the encoder, n is the filter width, lr is the encoder learning rate, d is the dropout, and m is the margin. For adaptive domain transfer, we also tune the domain classifier learning rate lr_2, the constant λ used in calculating loss, and the hidden dimensions of the domain classifier feed forward network h_2 and h_3. $|\theta_{dc}|$ is the number of parameters for the domain classifier.

Method	Dev				Test			
	MAP	MRR	P@1	P@5	MAP	MRR	P@1	P@5
BM25 (Lei et al., 2015)	52.0	66.0	51.9	42.1	56.0	68.0	53.8	42.5
CNN	55.7	69.7	57.1	44.0	55.7	69.7	57.1	44.0
LSTM	58.3	72.2	61.4	46.7	58.3	72.2	61.4	46.7

Table 2: Comparative results between the CNN and LSTM models on the question similarity task. Higher numbers are better. Mean pooling was used for both models. BM25 scores from (Lei et al., 2015) are reported for reference.

Method	Dev AUC	Test AUC
TF-IDF	0.707	0.739
Direct Transfer	TODO	TODO
Adv. Dom. Transfer	0.697	0.696
Exploration	TODO	TODO

Table 3: Comparative results between the CNN and LSTM models on the question similarity task.

Exploration We expand on domain adaptation techniques by using two separate neural encoders, one for the label predictor and one for the adversary. Learning in the adversarial domain adaptation model is highly unstable. We implement use an additional word-level auto-encoder reconstruction loss, inspired by (Zhang et al., 2017).

7 Experimental Setup

Dataset We train and evaluate the models on the Stack Exchange AskUbuntu dataset (Lei et al., 2015), which consists of 167K questions (title, body pairs) in total. To perform adversarial domain transfer, we use the Stack Exchange Android dataset (Guo, 2017), which consists a corpus of questions in the same format as above, to train the discriminator. We evaluate the models on the Android dev and test sets. The question bodies are truncated to a maximum length of 100 words.

Training, Development, and Test Sets For information retrieval, we exclusively use the AskUbuntu dataset. Each training set sample consists of the question q_i, a user-marked similar question p_i^+, and a set of 20 negative questions, which are drawn randomly from the entire corpus for each q_i. In theory, these questions may contain similar questions to q_i. However, due to the large size of the corpus, we expect that this is unlikely. The candidate set is $Q(q_i) = \{p_i^+\} \cup Q_i^-$. The development and test sets consist of questions, each with the top 20 similar candidates retrieved using BM25. The candidates are manually annotated as similar or non-similar.

For domain transfer, we use both the AskUbuntu and Android datasets. The training set consists of samples described above as well as samples consisting of 20 questions randomly chosen from the AskUbuntu corpus, 20 questions randomly chosen from the Android corpus, and their corresponding domain labels. Evaluation is performed on the provided Android dev and test sets which contain questions with positive and negative examples.

Baselines and Evaluation Metrics We report Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), Precision at 1 (P@1), and Precision at 5 (P@5) scores to evaluate the information retrieval models without domain transfer. We report the Area Under Curve (AUC) score for the domain transfer models. We compare our information retrieval results to those given in (Lei et al., 2015), and our adversarial domain transfer results to those of our baseline transfer model.

Hyper-parameters For the CNN, LSTM, and direct transfer models, we tuned the encoder learning rate, dropout (Hinton et al., 2012), hidden dimension, and margin. For the adversarial domain transfer model, we fixed the parameters of the best performing direct transfer model, and tuned the domain classifier learning rate, lambda, and hidden dimensions for the domain classifier feed forward network. We used Adam (Kingma and Ba, 2014) as the optimization method for all neural network models. The final set of parameters is shown in Table 1.

Word Vectors For the information retrieval part (without any domain transfer), we used the provided set of 200-dimensional pre-trained word vectors, trained from Stack Exchange and Wikipedia text. Since the domain transfer part introduces a new corpus vocabulary, we instead use off the shelf Glove (Pennington et al., 2014) embeddings. Both embedding files are pruned by removing words that are not found in the corresponding datasets for each part.

All models were implemented using PyTorch, version `0.2.0_3`. We used the Scikit Learn TF-IDF vectorizer for the TF-IDF model.

8 Results

The CNN and LSTM models beat the existing BM25 scores in every metric. Scores are shown in Table 2. The AUC scores for the TF-IDF, direct transfer, and adversarial domain transfer models AUC are reported in Table 3. In general, the LSTM model performed better than the CNN model. Thus, we used the LSTM encoder in the adversarial domain transfer task. The adversarial domain transfer model achieved a 12.0% relative increase in AUC score over the baseline (direct transfer model). We found that increasing dropout to 0.4 and using a larger λ value (effectively increasing the amount of regularization) was integral in improving performance.

9 Conclusion

We implement recurrent and convolutional neural models for encoding questions into semantic vector representations, which are then compared for the task of similar question retrieval in question answering forums. We also use domain adaptation techniques for the case where annotated data is readily available in one domain but not another. Although we beat the target scores for every model, we could not perform an extensive hyper-parameter search, explore different pooling strategies, or attempt using titles only in order to find the best models, due to time constraints. [1]

References

Y. Ganin and V. Lempitsky. 2014. Unsupervised Domain Adaptation by Backpropagation. *ArXiv e-prints* .

Jiang Guo. 2017. Android question dataset https://github.com/jiangfeng1124/Android.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR* abs/1207.0580. http://arxiv.org/abs/1207.0580.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980. http://arxiv.org/abs/1412.6980.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324. https://doi.org/10.1109/5.726791.

Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi S. Jaakkola, Kateryna Tymoshenko, Alessandro Moschitti, and Lluís Màrquez i Villodre. 2015. Denoising bodies to titles: Retrieving similar questions with recurrent convolutional models. *CoRR* abs/1512.05726. http://arxiv.org/abs/1512.05726.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

[1] The source code is located at `https://github.com/christinasun/6.864-Final-Project`. Instructions for running the final models described in this paper are found in the `README`.

Yuan Zhang, Regina Barzilay, and Tommi S. Jaakkola. 2017. Aspect-augmented adversarial networks for domain adaptation. *CoRR* abs/1701.00188. http://arxiv.org/abs/1701.00188.