6.864, Fall 2017 - Neural Approaches to Question Retrieval

Codebase: https://github.com/shashank-srikant/6.864_term_project

Vadim Smolyakov*
CSAIL, MIT
vss@csail.mit.edu

Shashank Srikant*
CSAIL, MIT
shash@mit.edu

ABSTRACT

Content on the internet grows at an exponential rate. Given this growth, finding relevant information accurately becomes a critical task for the NLP community to address. More so, with this rapid growth, curating labeled datasets to build models for the wide variety of content available on the internet has become extremely time and resource intensive. In this work, we explore whether neural models are able to successfully model content similarity and retrieval tasks, and whether they can transfer knowledge from one domain, where supervised labels are available, to a domain with no available labels. Specifically, we explore content similarity in online discussion forums, where we explore the following questions - a. how effectively can neural approaches model question-answer retrieval tasks, which is, given a question and answer pair present on an online discussion forum, how effectively can neural approaches find similar pairs on that forum. b. Given a model of question-answer similarity in one online discussion forum, how effective are neural approaches in transferring that knowledge to a new, loosely related domain. In this work, we explore a baseline approach of modeling question similarity tasks on the popular online community *AskUbuntu*. We show how neural architectures like LSTMs and CNNs outperform traditional approaches in information retrieval. Additionally, and importantly, we explore the problem of transferring these models to detect question-answer similarity on *Android stack exchange*, a similar yet different online discussion community which discusses Android related problems. We show how neural domain adaptation techniques successfully beat baseline IR techniques and direct neural transfer techniques. We also discuss some limitations and challenges in using such architectures.

1. INTRODUCTION

The problem of text similarity, and specifically, similarity of short queries or answers found on the internet, have been central to the modern NLP community. With the explosion in content on the internet, a lack of robust tools to find similar content has the risk of creating further similar and redundant content, which only exacerbates the original problem.

Another relevant and pressing concern which such an explosion and variety of content has introduced is the increased cost of building predictive NLP, NLU models which cater to such content. The variety in content requires building a model from scratch, irrespective of how closely related the content may have been to a previously built model. For instance, if we were to model the reviews written for movies, we would have to reinvest effort and time in modeling reviews written for another domain, say, food or hotels. In spite there being conceptual, semantic similarities between the tasks of reviewing movies and reviewing food, models have to be created anew. And each such modeling exercise demands a large repository of curated, preferably labeled data, which most often is not feasible to put together. The pressing challenge such a variety of content has created is to be able to learn with minimal supervision, and from loosely related datasets.

In this work, we investigate two problems - one, to model question-similarity tasks using state of the art techniques in neural modeling and two, transfer those models to a loosely related, yet equally rich domain under the constraint of having no supervised information on that new domain. Specifically, we explore the problem of finding similar question-answer pairs on the popular online discussion *AskUbuntu*.[1] This discussion forum focuses on troubleshooting queries on Ubuntu, the popular open-source operating system. In addition to learning models to find relevant question-answer pairs on this forum, we also use this domain to transfer knowledge onto modeling question-answer similarity in *Android stack exchange*.[2] This is an online community, similar in setup and structure to *AskUbuntu*, wherein short queries and corresponding answers on Android-related troubleshooting are present.

In this work, we investigate how state of the art neural architectures for NLP applications perform on the task of question similarity. Traditionally, text similarity tasks were solved using information retrieval techniques like building an indexer and applying ABC and ABC, which consider XYZ. We study how more recent techniques like RNNs and CNNs, which have shown to successfully model text on tasks like XYZ, model this particular task. For the domain adaptation task of learning question similarity in Android stack exchange, we explore adversarial neural techniques, and Doc2Vec [**?**] in learning from the *Ask Ubuntu* dataset.

The study demonstrates the relevance of neural techniques in a critical NLP task like question-answer retrieval. We demonstrate how using neural approaches, one can outper-

*Author order decided by tossing a fair coin.

[1] https://askubuntu.com

[2] https://android.stackexchange.com

Scenario	Setting	Nature of Data	Learning Paradigm	Main Concepts
$\mathcal{D}_S = \mathcal{D}_T$, $T_S = T_T$	Traditional Machine learning	Labelled data in source domain(s) and unlabeled data in target domain	Source and target domains are exactly the same	Learn models on training set and test on future unseen data
$\mathcal{D}_S \neq \mathcal{D}_T$, $T_S = T_T$	Transductive Transfer Learning	Labelled data in source domain(s) and unlabeled data from $\mathcal{P}(X_S) \neq \mathcal{P}(X_T)$	Single source domain adaptation	Learning common shared representation; instance weighing, parameter transfer
			Multi-source adaptation	Classifier combination; efficient combination of information from multiple sources; Feature representation
No conditions on $\mathcal{D}_S$, $\mathcal{D}_T$, but $T_S \neq T_T$	Inductive Transfer Learning	Unlabeled data in source domain(s) and labeled data in target domain	Self-taught learning	Extracts higher level representations from unlabeled auxiliary data to learn instance-to-label mapping with labeled target instances
		Labeled data is available in all domains	Multi-task learning	Simultaneously learns multiple tasks within (or across) domain(s) by exploiting the common feature subspace shared across the tasks
$\mathcal{D}_S \neq \mathcal{D}_T$ $T_S \neq T_T$	Kim et al., 2015 [13]	Labeled data in source and target domains	Transfer learning with disparate label set	Disparate fine grained label sets across domains, however, same coarse grained labels set can be invoked across domains

Table 1: A brief summary of different transfer learning approaches. The cell marked in blue highlights the approach explored in this work. This summary is based on the work detailed in the survey paper by Pan et. al.[16]

form traditional information retrieval techniques while not having to invest heavily in engineering the right features to get to such a performance. Additionally, we show the successful transfer of domain knowledge from one domain to another using state of the art neural transfer techniques.

This work is organized into the following sections - Section 2 discusses related work in this field. Sections 3, 4, 5 discuss the various techniques to model in-domain and domain adaptation tasks for the given problem. Section 6 discusses the experiment setup and the results. Section 7 discusses the results and concludes our work.

2. RELATED WORK

Given the growing popularity of community QA forums, question retrieval has emerged as an important area of research. Recent work has also gone beyond word-based methods to represent this task. For instance, Feng et. al. [7] learn word embeddings using category-based metadata information for questions. They define each question as a distribution which generates each word (embedding) independently, and subsequently use a Fisher kernel to assess question similarities. Dos Santos et. al. [6] propose an approach which combines a convolutional neural network (CNN) and a bag of words representation for comparing questions. In contrast to [7], Lei et. al. [14] treat each question as a word sequence as opposed to a bag of words, and apply recurrent CNNs instead of traditional CNNs. We base our work on [14]. We apply different off-the-shelf deep learning models to encode a meaningful embedding and define similarity over these embeddings.

Domain adaptation has been an active area of research in NLP. Table 1 summarizes different settings of transfer learning [16]. Our work explores techniques in unsupervised domain adaptation, which operates when the source and target domains ($\mathcal{D}$) are different, but share a common task (T_i) to achieve. It helps understand where this approach lies in the spectrum of techniques.

We study in this work the scenario where the domains vary while the tasks remain the same. This is referred to as transductive transfer learning. This is the most extensively studied settings in the transfer learning literature and can be broadly categorized as single and multi-source adaptation. Single source adaptation[4; 1; 5] primarily aims at minimizing the divergence between the source and target domains either at instance or feature levels. The general idea being identifying a suitable low dimensional space where transformed source and target domains data follow similar distributions and hence, a standard supervised learning algorithm can be trained.

Multiple methods perform unsupervised domain adaptation by matching the feature distributions in the source and the target domains. Some approaches perform this by reweighing or selecting samples from the source domain [3; 12; 10], while others seek an explicit feature space transformation that would map source distribution into the target ones [15; 11; 2]. An important aspect of the distribution matching approach is the way the (dis)similarity between distributions is measured. Here, one popular choice is matching the distribution means in the kernel-reproducing Hilbert space [3; 12], whereas [10; 8] map the principal axes associated with each of the distributions. The approach explored in this work is based on [9] and also attempts to match fea-

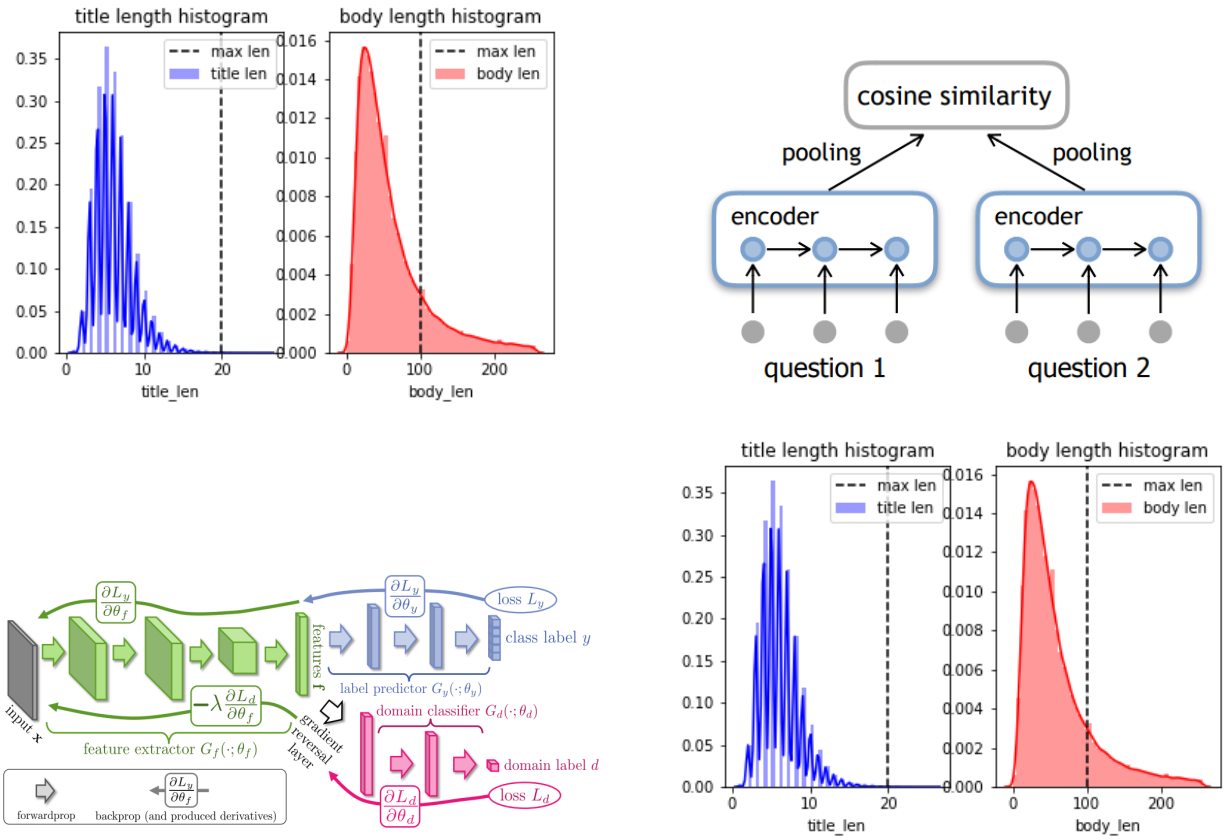

Figure 1: Summary of the *Ask Ubuntu* dataset

ture space distributions, however this is accomplished by modifying the feature representation itself rather than by reweighing or geometric transformation. Also, the method used to measure the disparity between distributions based on their separability is implicitly different and is done by a deep discriminatively-trained classifier.

3. IN-DOMAIN QUESTION SIMILARITY

We first explore the task of question similarity by training a domain-specific classifier. The domain we have explored in this work involves the questions and answers asked on *AskUbuntu*, a popular online discussion forum where the community troubleshoots problems related to Ubuntu, the operating system. The dataset for this task was curated by [14].

	source($\mathcal{D}_S$)	target($\mathcal{D}_T$)
Description	*Ask Ubuntu*	*Android SE*
# of sentences	40	50
# of stop words	50	90
mean word length - title	299	300
mean word length - body	299	300

Table 2: Summary statistics of our dataset

3.1 Dataset

We describe here the dataset that we base our work on.

- The *Ask Ubuntu* dataset consists of question and answers related to problems people face in Ubuntu, the operating system
- Each query has two components - a title and main body. The average title length is XX words and the average body length is YY words long. Figure 1 shows the distribution of word lengths.
- In this work, to limit computational requirement, we consider only the first XX words in the title and the first YY words in the body of each sentence.
- The *Ask Ubuntu* platform has a section where users flag similar questions. We use that as a weak signal to inform our true labels, since these user-flagged suggestions are not complete and suffer from high recall issues.
- In order to train our classifiers with dissimilar pairs of questions, each query is also associated with randomly drawn queries from the sample.
- We trained on roughly 110,000 query-answer pairs to train our in-domain classifier.
- In the test set, for each pair of similar and dissimilar questions, we are also given the BM25 scores, their similarity as measured by an industrial IR-system. These serve as a baseline for all our experiments.

3.2 Model

We model the task of in-domain question similarity as follows:

- We define a word-level deep neural network. The network takes in a word at a time, and is represented by embeddings. This word can either belong to the title or the body of a query. In this work, we average the embeddings from the title and body of each query, and use the averaged embeddings to classify similarity.
- We use pre-trained embeddings on data drawn from both, the source domain and Wikipedia. Both these data sources are rich, are easily available and training these embeddings using Word2Vec etc. are relatively straightforward.
- To train the classifier to find similar and dissimilar questions for a given query, we associate each query with two sets of texts - the set of similar questions as marked by users on *Ask Ubuntu*, and 100 randomly sampled questions from the corpus which serve as dissimilar pairs. The user-defined questions are used only during training. Since user-marked similar questions are not comprehensive and serve only as a weak proxy, the system is tested on expert-marked similar questions.
- In this setup, we find the distances between a query and its similar questions and dissimilar questions by defining a similarity operation over its embeddings. We use the cosine distance between the embeddings as a similiarity operation.
- We use a maxmargin framework for learning parameters θ of our network. Specifically, in a context of a particular training example where q_i is paired with p_i^+, we minimize the max-margin loss $L(\theta)$ defined as

$$\max_{p^- \in Q(q_i)} \{s(q_i, p^-; \theta) - s(q_i, p_i^+; \theta) + \delta(p, p_i^+)\} \quad (1)$$

  where $\delta(.,.)$ denotes a non-negative margin. We set $\delta(p, p_i^+)$ to be a small constant when $p \neq p_i^+$ and 0 otherwise. The parameters θ can be optimized through sub-gradients $\partial L/\partial\theta$ aggregated over small batches of the training instances [14].
- The neural network essentially serves as an encoder to generate a representation in a low-dimension space which encodes the loss function $L(.)$ defined above.
- In this work, we implement two neural networks, LSTMs and CNNs, as these encoders.

CNN		LSTM	
Parameter	Value	Parameter	Value
# of sentences	40	# of sentences	50
# of stop words	50	# of sentences	90
mean word length - title	299	# of sentences	300
mean word length - body	299	# of sentences	300

Table 3: Parameters of our encoder networks for in-domain classification

3.3 CNN

We describe the CNN architecture implemented in this work. We detail the inputs to the different layers of the network, which should help illustrate the architecture. We define a CNN with the following layers -

- **Input vector X** of dimensions: (**B**, **W**) (**B**: batch size, **W**: max # of words in sentence in batch)
- **Embedding layer** Pass **X** through embeddings layer. Resulting dimension: (**B**, **W**, **D**) (**D**: # of embeddings per word)
- **Convolution, ReLU** Pass the above input through a convolution layer and then through ReLU. Resulting dimension: (**B**, **K**, **Y**)(**K**: Depth of filter, i.e. # of filters of given height, **Y**: # of values sliding the filter over the input)
- **Pooling** Pass through mean pooling. Resulting dimension: (**B**, **K**)
- The resulting encoded embedding of dimension (**B**, **K**) for **B** words in a given batch.

3.4 LSTM

We describe the CNN architecture implemented in this work. We detail the inputs to the different layers of the network, which should help illustrate the architecture. We define a CNN with the following layers -

- **Input vector X** Dimensions: (**B**, **W**) (**B**: batch size, **W**: max # of words in sentence in batch)
- **Embedding layer** Pass **X** through embeddings layer. Resulting dimension: (**B**, **W**, **D**) (**D**: # of embeddings per word)
- **Convolution, ReLU** Pass the above input through a convolution layer and then through ReLU. Resulting dimension: (**B**, **K**, **Y**)(**K**: Depth of filter, i.e. # of filters of given height, **Y**: # of values sliding the filter over the input)
- **Pooling** Pass through mean pooling. Resulting dimension: (**B**, **K**)
- The resulting encoded embedding of dimension (**B**, **K**) for **B** words in a given batch.

4. DOMAIN ADAPTATION (DA)

We use neural adversarial techniques[9] to adapt to a domain having no labeled data, having trained a model on a domain with labeled data. In this section and the next, we describe the dataset we worked on, the architecture for the adversarial model we use, and other models for DA we tried.

4.1 Dataset

We worked with two datasets, one as our source domain, wherein we had labeled data and another as our target domain, where we had no labeled data. The task on both these datasets was the same - to predict similarity between questions. The source domain in our experiments was the *Ask Ubuntu* dataset, described in the previous section. The target domain that we wanted to transfer this model to was the *Android stack exchange*, a question-answer site on Android-related issues. Table 2 describes the summary statistics of this dataset. For the target domain, we had XX sentences to test. Each query was expert-annotated with similar and dissimilar queries they corresponded to. No BM25*scores* were provided to evaluate a baseline. Table XX describes the architecture of the encoders and decoders.

4.2 Model

We model this task of unsupervised domain adaptation using a neural adversarial approach [9]. The core idea is as follows -

- We define an encoder which is able to take in the data from the source domain and learn a representation which predicts the similarity for tasks in its own

CNN		LSTM	
Parameter	Value	Parameter	Value
# of sentences	40	# of sentences	50
# of stop words	50	# of sentences	90
mean word length - title	299	# of sentences	300
mean word length - body	299	# of sentences	300

Table 4: Parameters of the encoder and domain differentiator networks for domain adaptation

domain, given that we have labels for this domain. This is similar to the in-domain learning described in the previous section.

- While the primary goal of predicting $\mathcal{Y}_{\text{source}}$ remains, we define an adversarial objective to guide the direction of the encoded embeddings in a way to make them abstract enough to be able to discriminate $\mathcal{Y}_{\text{target}}$ as well.
- We do this by setting an objective in the adversarial to predict the label source or target. If the origin of the input data is being discriminated well by the adversarial network, a negative weight is propagated to the encoder, thereby making the encoding abstract enough to not being able to distinguish between the two domains, while being able to predict $\mathcal{Y}_{\text{source}}$ well. This ensures the successful transfer of the encoded embeddings to the target domain. Figure 4.2 depicts this arrangement.

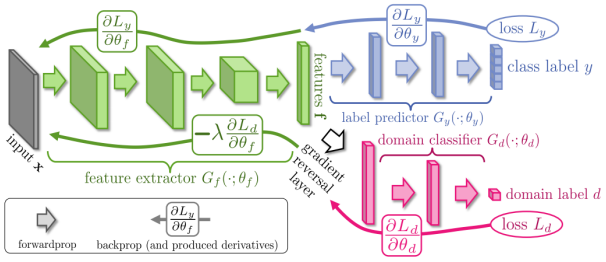

Figure 2: Adversarial network for unsupervised domain adaptation investigated in this work. Image reference: Ganin et al., 2015 [9]

5. OTHER TECHNIQUES FOR DA

6. EXPERIMENTS

7. DISCUSSION

8. ACKNOWLEDGEMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the **.cls** and **.tex** files that it describes.

9. REFERENCES

[1] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(Nov):1817–1853, 2005.

[2] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann. Unsupervised domain adaptation by domain invariant projection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 769–776, 2013.

[3] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

[4] B. Chen, W. Lam, I. Tsang, and T.-L. Wong. Extracting discriminative concepts for domain adaptation in text mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 179–188. ACM, 2009.

[5] H. Daumé III. Frustratingly easy domain adaptation. *arXiv preprint arXiv:0907.1815*, 2009.

[6] C. Dos Santos, L. Barbosa, D. Bogdanova, and B. Zadrozny. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 694–699, 2015.

[7] M. Feng, B. Xiang, M. R. Glass, L. Wang, and B. Zhou. Applying deep learning to answer selection: A study and an open task. In *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*, pages 813–820. IEEE, 2015.

[8] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Proceedings of the IEEE international conference on computer vision*, pages 2960–2967, 2013.

[9] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189, 2015.

[10] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2066–2073. IEEE, 2012.

[11] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 999–1006. IEEE, 2011.

[12] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola. Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems*, pages 601–608, 2007.

[13] Y.-B. Kim, K. Stratos, R. Sarikaya, and M. Jeong. New transfer learning techniques for disparate label sets. In *ACL (1)*, pages 473–482, 2015.

[14] T. Lei, H. Joshi, R. Barzilay, T. Jaakkola, K. Tymoshenko, A. Moschitti, and L. Marquez. Semi-supervised question retrieval with gated convolutions. *arXiv preprint arXiv:1512.05726*, 2015.

[15] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.

[16] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.