Learning Question Similarity in CQA from References and Query-logs

Alex Zhicharevich[1], Moni Shahar[2] and Oren Sar Shalom[1]
[1]*Intuit AI, Israel*
[2]*Facebook, Israel*

Keywords: Community Question Answering, Text Similarity, Text Representation, Deep Learning, Weak Supervision.

Abstract: Community question answering (CQA) sites are quickly becoming an invaluable source of information in many domains. Since CQA forums are based on the contributions of many authors, the problem of finding similar or even duplicate questions is essential. In the absence of supervised data for this problem, we propose a novel approach to generate weak labels based on easily obtainable data that exist in most CQAs, e.g., query logs and references in the answers. These labels accommodate training of auxiliary supervised text classification models. The internal states of these models serve as meaningful question representations and are used for semantic similarity. We demonstrate that these methods are superior to state of the art text embedding methods for the question similarity task.

1 INTRODUCTION

Community Question Answering (CQA) sites have emerged as a popular and rich source for information, where both domain specific sites (e.g., StackOverflow[1], TTLC[2]), and general purpose sites (e.g., Quora[3]), provide a platform for users to get precise answers to their questions. Though community contribution is crucial in order to keep such huge amount of information up to date, question authors are not necessarily aware of all the already existing questions. It is therefore common to find similar or even duplicate questions. A question to question similarity measure is important in many applications like question deduplication, similar question recommendation, question routing, retrieval and more (Nakov et al., 2017; Srba and Bieliková, 2016).

While each CQA site may have unique traits, all major sites share the following three components: *structure* of questions, a *reference mechanism* and an internal *search engine*. In detail, the structure of a question is composed of a mandatory *title* and an optional *details* section. The title is usually one or two sentences containing the subject of the question. The details is a short section, few sentences long, that elaborates on the question. An answer to a question is termed as a *reply*. The *reference mechanism* allows to link within a reply of a *seed* question to a *referenced* question.

Most existing approaches for CQA question similarity require labeled data (Nakov et al., 2017). However, in many real world applications this supervision does not exist and it is impractical to obtain a large corpus of labeled data. Our proposed approach leverages auxiliary data to generate abundant amount of weak supervision. Then we apply a neural network to learn representations for questions, which in turn would be used to calculate question similarity.

2 RELATED WORK

The huge amount of data residing in CQAs have attracted a significant amount of attention from the research community. Multiple aspects of CQA systems have been studied (Srba and Bieliková, 2016), such as question and answer retrieval, automatic question answering, question quality and more. Question to question semantic similarity is a fundamental task and multiple approaches have been tried to model it. One of the subtasks in SemEval CQA task (Nakov et al., 2017) was directly aimed at finding good question to question similarity measures. The task was structured as a supervised retrieval problem, imposing that the majority of the proposed solutions modeled the similarity using unsupervised text similarity methods which are later combined to a similarity score us-

[1]https://stackoverflow.com
[2]https://ttlc.intuit.com/
[3]www.quora.com

Zhicharevich, A., Shahar, M. and Shalom, O.
Learning Question Similarity in CQA from References and Query-logs.
DOI: 10.5220/0008982403420352
In *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2020)*, pages 342-352
ISBN: 978-989-758-397-1; ISSN: 2184-4313

ing a supervised classification method. (Charlet and Damnati, 2017) which were the top system at SemEval17, used a supervised combination of SoftCosine similarity over question tokens with cosine distance between question embeddings using word embedding weighted average. (Nassif et al., 2016) used stacked bidirectional LSTM to learn similarity classes between questions. (Shao, 2017) proposed learning the element wise differences of the outputs of CNN to generate similarities. (Filice et al., 2017) trained an SVM over engineered intra and inter pair similarity features. Similar approaches were proposed to capture general semantic similarities between texts in SemEval STS (semantic text similarity) task (Cer et al., 2017) as well as for a more specific deduplication task (Bogdanova et al., 2015; Zhang et al., 2017). Two major limitations of those methods are high complexity and the need for labeled datasets to train the final supervised component.

Question similarity can be considered as a special case of a general text similarity problem. A popular approach for text similarity is to encode the text in low dimensional vector spaces which capture the text semantics and later model the similarity as a geometric proximity in the embedding space. One approach to produce sentence level embeddings is by composing word level embeddings (Arora et al., 2017; Pagliardini et al., 2018). Alternative modern approaches use sentence level tasks to directly encode longer texts. (Kiros et al., 2015) proposed such an embedding by trying to reconstruct the surrounding sentences of the encoded sentence and (Logeswaran and Lee, 2018) structured this idea as a classification problem. (Conneau et al., 2017) train a universal sentence embedding method using supervised methods on the SNLI dataset. (Cer et al., 2018) use Transformer and DAN networks to produce general embeddings that transfer across various tasks. Though most of these text embeddings methods are trained to be successfully applied on a variety of downstream tasks they are not optimized directly for pairwise similarity. Pretrained language models (Devlin et al., 2019) are successfully improving results on many NLP tasks, but are designed to be fine-tuned with labeled data and are not meant to serve as sentence representation extractors.

In contrast to general text representation approaches, some work has been done to learn representations that directly optimize the similarity task. Siamese networks (Chopra et al., 2005) are a popular framework to learn similarities and have been used for text by (Mueller and Thyagarajan, 2016) and (Neculoiu et al., 2016). A more general triplet architecture (Hoffer and Ailon, 2015) was applied for text similarity by (Ein Dor et al., 2018). While these methods provide a principled way of modeling similarity directly between arbitrary texts, where the architecture of the sub-network can be adapted to the texts at hand, they do not aim to be transferred as is to other domains and require a substantial amount of labeled data as well as sophisticated negative sampling methods.

Unlike NLP methods, which measure similarity between questions using the question's content, methods from information retrieval have a long history of inferring such similarities from the click through data recorded by search engines. This line of work uses user clicks on query results as an implicit relevance signal (Baeza-Yates and Tiberi, 2007) and thus queries can be represented as a function of the clicked documents and vice versa. Popular approaches suggest to construct either a bipartite graph of queries and documents and leverage its structure to learn similarities (Craswell and Szummer, 2007; Jeh and Widom, 2002), or an equivalent sparse query-document click through matrix. Models leveraging deep learning were also proposed (Huang et al., 2013) for this task. A majority of the existing work focuses on query similarity (Ma et al., 2008), but some suggestions were made (Poblete and Baeza-Yates, 2008; Wu et al., 2013) to apply these methods for documents as well. While some work was done on query log analysis of CQA (Figueroa and Neumann, 2013; Wu et al., 2014) we are not aware of any in the context of question similarity. A major limitation of using strictly click through information, is the existence of questions for which the number of recorded clicks is very limited or even non existing. This is natural for newly posted questions, and in our case also frequent for many existing unpopular questions

3 THE DATA - TurboTax LIVE COMMUNITY

This work focuses on question similarity on the TurboTax Live Community (TTLC) site. TurboTax, developed by Intuit, is the most widely used tax filing service in the US. TTLC is a tax related CQA operated by Intuit that provides a wide knowledge base for US tax filers, containing over 3 million questions with a mix of tax related and product related questions. As described above, questions are usually posted with question phrased titles describing the general information the user looks for and a details section that describes the specifics of the posting user's question. Like in many other CQA platforms, question are usually answered by domain experts like tax profession-

als or by the Intuit TurboTax support representatives who are well versed in the details of the TurboTax product. Since most posting users are looking for factual information on tax regulations or product guidance, the vast majority of questions result in a single answer providing the needed information.

Although the tax system is complex, the breadth of topics on TTLC is much smaller compared to platforms like Quora or StackOverflow. The vast majority of posts relate to a relatively small set of central topics either around tax regulations or the operation of the TurboTax product with different variations dependant on the posting user specific situation. To address this need, Intuit is maintaining several thousands of high quality expert-created guide pages (FAQs) which provide general high quality information around those central topics. Due to this nature, a common case is that a reply to a question will be a reference to another answer or more frequently an FAQ page, and TTLC's user interface allows the rendering of the referenced page within the original reply which provides a citation-like interface to those references.

TTLC's search engine records all queries submitted to it from users, as well as all questions clicked following those queries. Due to the varying quality of questions on the site, the search engine is designed to rank the higher quality questions higher, especially Intuit generated FAQs. This results in many community generated questions receiving very low traffic. A common scenario for a posted question is to be answered after surfacing in the unanswered questions queue, be read by the posting user after he was notified it was answered, and later never be clicked by other users as a search result (though users can be exposed to questions in ways other than search). Our modeling decisions were therefore impacted by the high volume of questions not appearing in query logs.

4 PROPOSED MODEL

With the absence of massive amount of labels required for training modern supervised models, our solution is divided into two parts: (i) learn question representation based on weak supervision and (ii) apply the representations to measure similarity.

We propose two methods for learning question representation using two auxiliary sources of information: the *reference mechanism* and *search engine*. Both methods leverage the auxiliary information to construct a classification training set, where the question is used to predict its properties - the reference in the question reply or the query that was used to retrieve the question. With these training sets, a neural network classifier is trained, and the question representation is obtained by extracting the last layer of the network. Later, we use the fact that the two models share similar formulations to train a joint model that leverages both the queries and references.

4.1 Reference Prediction Model

The reference prediction model uses the assumption that similar questions have similar replies. However, reply texts are usually of even greater complexity than those of the questions. Two replies that reference the same question usually resolve around the subject of the referenced page, thus indicating that the corresponding question are similar as well. The large amount of references present in the TTLC system allows us to simplify and use the second assumption to learn similarity. Table 1 shows random examples of questions and the title of their referenced question. As this table shows, the content of the reference is indicating the topic of interest of the original question. We therefore treat the referenced page as a weak signal for the topic of the referencing question. Note that while in TTLC the referenced pages are also part of the same CQA, this is in general not required for our algorithm, since the content of the referenced page is not used by the model. In other domains the references can point to an arbitrary content such as Wikipedia pages.

In order to find questions with references, a regular expression search over the replies text was performed to extract the references, resulting in a dataset of $210,308$ questions. In the case of several references in the same reply or in the case of multiple replies for the same question, we took the first reference in the chronologically first reply. We then frame the task of reference prediction as a multi-class text classification, where each unique reference is represented with a class. As one may expect, the frequency of referenced pages is highly skewed, with a small number of highly referenced pages and a long tail of pages referenced only once. While training can be done with such configuration, this is sub optimal both due to the low number of training examples per class as well as the potential duplication between the referenced pages content. To address this limitation, we restrict the number of unique referenced pages in the training set, requiring at least 20 referencing pages per reference. This resulted in 893 referenced pages in the dataset and $162,468$ question-reference pairs.

The classification model used neural network, and followed popular configuration (İrsoy and Cardie, 2014) for text classification. It contained the following four parts.

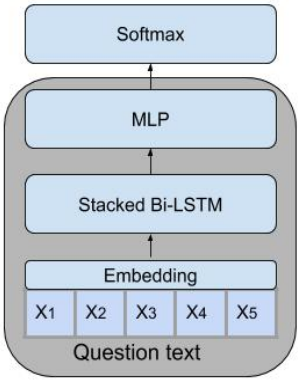

Figure 1: Neural Network architecture of reference prediction model.

- Learn a domain specific vector representations for all words in the titles of the questions. Representations are learned with the Word2Vec SGNS embedding scheme (Mikolov et al., 2013). This is done in a separate pre-processing step.
- Embed the question titles in the training set by representing words with the learned embeddings.
- Process the embedded question titles with two stacked bidirectional LSTM layers (Schuster and Paliwal, 1997; Hochreiter and Schmidhuber, 1997). The concatenation of the last hidden states of the forward and backward parts of the second LSTM serve as the encoding of the question title to a fixed size vector representation.
- The question representations are fed into two fully connected layers and softmax normalization. The output of the model is a probability distribution of a question to reference each of the target references.

The model architecture is presented in details in Figure 1. The highlighted part of the network is the representation extractor, namely we represent each question as the output of the last hidden layer of the MLP part. Naturally, after the network is trained representation can be obtained for arbitrary questions, not restricted to those appearing in the model training set.

4.2 Query Log Prediction Model

The query log is a valuable source of information for similarity. In this approach the training data are the query logs of the CQA, namely, the questions that were clicked following each query. While the reference approach assumed similarity of two question based on sharing the same reference in the reply, here our modeling is based on the idea that questions are similar if they are viewed by users that searched for the same queries. The query logs of TTLC are query-question pairs (q_i, d_j) which are the result of a user entering the query q_i in the CQA search engine, and clicking question d_j from the search results. As described in Section 3, a significant portion of questions has very little or even no click-through data. This is naturally the case for newly posted questions. Therefore, though it is possible to compute similarities between documents directly on the click-through pairs, we use the click-through data again as weak labels set for a training set for a text based classification model.

By grouping the click-through pairs list P, we construct a set $L_{top} = \{q_1 \dots q_n\}$ of the n most popular queries, where from early experiments we set $n = 2000$. We then filter P to contain only pairs (q_i, d_j) where $q_i \in L_{top}$. After the filtering we remained with $\approx 500K$ pairs where the number of unique questions is over $25,000$. We enumerate the queries and then train the exact same neural network as in 4.1 where for a click pair (q_i, d_j) the query index of q_i is predicted using the title of question d_j. An important difference from the reference model is the fact that unlike in the reference model, where we select just one reference per question, in this model the training data may contain several different labels for the same question. The click counts were used only for filtering purposes, and after the filtering stage no grouping or counting is performed on the click data, i.e., each click is a distinct training example.

4.3 Combining Reference and Query Log Data

Sections 4.1 and 4.2 propose different ways to construct weak labeling for learning question's representations. Due to the different data which is fed to each of the methods, each representation encodes different properties of the questions. The semantic generalization of the query log model is limited by the retrieval function of the CQA search engine, while the reference model may present poorer performance on topics where high quality references do not exist. Therefore it is useful to combine the two methods to get a representation which fuses both types of information. One straight forward way of combination is by vector operation like concatenation or averaging. A more sophisticated alternative is to jointly learn both supervised tasks from the question's content. We experiment with two approaches for such joint learning.

Multi-task learning (Ruder, 2017) is a popular approach in which a single model tries to reconstruct two targets based on the same input. In our case, this can be applied by trying to predict both the reference and the query. However, this is not straight forward

Table 1: Example of questions and referenced pages.

Question title	Referenced page title
some how i clicked I am self employed and we are not and now its asking for that info. Where or how can I delete this or should I put a zero in those spots	Can I downgrade to a lower-priced version of TurboTax Online?
I get a refund amount thats less than what was on TurboTax???	Why is my federal refund less than I expected?
I need to file taxes for my invalid daughter. How can I start a new income tax without errassing min	How do I start another return in TurboTax Online?
Why wont it take my credit card for IRS payment. I have been trying all day.	Can I pay my IRS taxes with a credit or debit card?

since in the query log training set, the same question can appear with multiple queries as labels unlike a single reference in the reference model. On top of that, the intersection of questions both appearing in the query logs and have references in the reply is relatively low. Since we also want to restrict the number of queries the model predicts, this will result in only several thousands question with both targets. Therefore we propose two alternatives to the straight forwards multi-task setting. The first, instead of framing the query prediction as a multi-class classification problem and minimizing the cross entropy between the predicted and actual classes, we will have the network, on top of the reference prediction, also minimize the dot product between the question's representation and the query representation where we represent a query with the average embedding of it's tokens. We do this for all query-click q_i, d_j pairs where d_j has a reference in one of it's replies. This method is referred in the results as the multi-task method and is illustrated in Figure 2 with the highlighted part representing the component later used for extracting question representations. The second method takes quite a different approach, in which we do not change the supervision framing setting. Instead, we reuse the same weights for learning both tasks, where we train the models in an alternating manner. We share all layers of the network except the last fully connected layer and the Softmax normalization (which is exactly the question encoding part), and train the model for an epoch with the reference target and then an epoch with the query target. We do this alternating training for 8 iterations. We later use the shared part of the two networks as a question encoder and refer to this combination as the iterative method.

4.4 Modeling Question Details

Oftentimes users ask very specific question which cannot be phrased using one or two sentence long titles. Specifically, in the tax domain the user may need to describe the precise details of his case to get an accurate answer. Due to this fact, the text contained in the details section may be very important for similarity estimation. However, due to its length and its complex structure it is harder to process. Since the lion's share of information still resides in the question titles, we compare several methods to include the details in the similarity calculation along with the title.

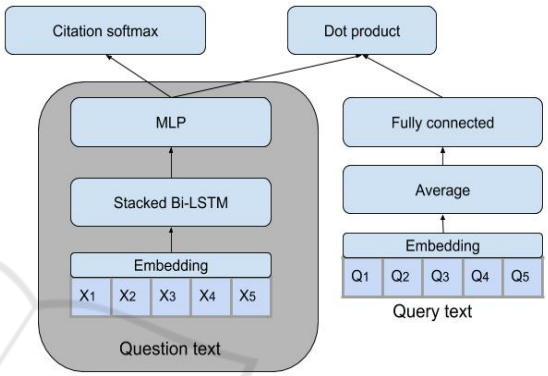

Figure 2: Neural network architecture of the multitask model.

4.4.1 Unsupervised Combination

The simplest way to combine the two parts of the text is by concatenating the details and the title into one text and apply the methods from previous sections. Though compelling for it's simplicity, this method ignores the semantic differences between the title and the details. To overcome this problem we combined the representations instead of combining the text. We separately encode the title and the details using any of the methods previously described so as to obtain a separate representation for each part. Once we had this representation we combined the two representations using either averaging or concatenation.

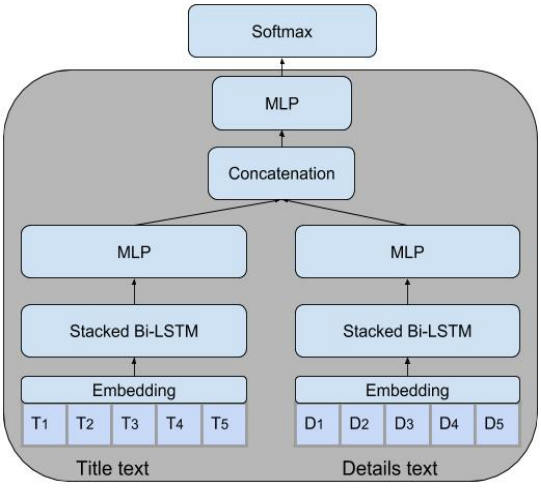

Figure 3: Neural network architecture of jointly predicting references using title and details.

4.4.2 Joint Learning

In order to avoid the complication in the previous section, we learned the problem on both the title and the details jointly. Unlike the concatenation of the text, separate networks are used to process each of the title and the details, and the layer outputs of the networks are concatenated, and serves as an input to several additional layers for the prediction. In theory the same network can be used to process the two texts with shared weights, but this will prevent the network from addressing the structural differences between the two texts, for example their length. Even without weights sharing, the simple network architecture used for titles can be inadequate for the details section which is often much longer. We experiment with two models for joint learning: (i) symmetric independent model and (ii) hierarchical model. In the symmetric model two identical networks which structure is described in 4.1 are learned with no weight sharing, so the first network encodes the title and the second encodes the details. Each network outputs a vector of dimension 64 and the concatenation of both vectors enters another fully connected layer so as to fit the labeled output. The network is illustrated in Figure 3 .The second model uses hierarchical attention networks (HAN) (Yang et al., 2016) for the encoding of the details, and is similar to the first model in all remaining components. This special configuration was designed to address the problem of the long text in the details.

5 EXPERIMENTAL RESULTS

5.1 Evaluation Protocol

Measurement of quality of a similarity function is somewhat hard since there are multiple traits that we expect from a good similarity function. Possible ways to do that are either to discretize the similarity to fixed classes (Conneau et al., 2017) or to let the similarity function to retrieve the nearest neighbours of a point and measure the resulted set in IR based measures (Nakov et al., 2017). Classification between related vs. unrelated question may be trivial in some cases and classification of similarity to fixed classes may be highly subjective. In this work we measure similarity using relative comparisons of two similarity scores. We manually created a set of triplets of questions (q, q_p, q_n), such that the similarity between q and q_p (positive) is higher than q and q_n (negative). Since compiling such a set is like looking for a needle in a haystack, we assist with the TTLC tagging system, which allows users to assign tags to questions. We manually composed a set of ≈ 400 tags which represent clear important concepts around the tax domain. We then sampled pairs of questions that have 2 common tags. Those questions act as candidates for (q, q_p), and after manually filtering pairs with a low semantic relatedness we ended up with $1,174$ pairs with a solid connection. We have two methods to introduce the negative question q_n as to generate the desired (q, q_p, q_n) triplets: *coarse-grained* that randomly selects a question with no common tags with q and *fine-grained*, that randomly selects a question that has a single common tag with q. The latter approach finds negative samples with some relatedness to q, making the differences more subtle. A sample of the fine grained similarity dataset is presented in Table 2 and from the coarse-grained dataset in Table 3.

5.2 Experimental Settings

5.2.1 Evaluation Measures

Given a similarity function *sim* and a triplet $t = (q, q_p, q_n)$, we define that *sim* is correct on t by

$$I_{sim,t} = \begin{cases} 1 & sim(q,q_p) > sim(q,q_n) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Table 2: Example question triplets from the fine-grained similarity dataset.

Seed question	First reference question	Second reference question
Requested the MAC software to amend my 2016 taxes. Turbo emailed a link with a code and password.	need to amend 2016 but don't remember my password	do I need to send a copy of my amended federal tax return with my Louisiana state return
what if I don't have my 1095	how do i revise self employed health insurance figure once entered?	With the cost of medications and healthcare, does HIV qualify as severely disabled?

Table 3: Example question triplets from the coarse similarity dataset.

Seed question	Reference question	Random question
How to deal with an overfunded 401k.	If a 401K contribution includes cents, and the tax contribution is rounded does that have any future tax implication?	Does being enrolled in health in 2015 qualify if it began in 2016?
how do we file jointly?	Divorce not final on Dec 31, can I file single?	Was last years taxes filed?

For a dataset D of n triplets, the ratio of correctly classified triplets by *sim* is simply

$$\frac{\sum_{i=1}^{n} I_{sim,t_i}}{n}$$

We denote this ratio as similarity accuracy. For each method we report the similarity accuracy for both the fine-grained and coarse datasets. As explained in the previous section, in each triplet t, q and q_p are the same in both datasets and q_n is random in the coarse dataset and shares a tag with q in the fine-grained dataset

5.2.2 Baselines

We consider a variety of methods for similarity computation as baselines. We mix methods that directly address similarity, as well as text representation methods. For text representation we include both models that use unsupervised word vector combination as well as pre-trained models trained on other domains. All our models and baselines use the same tokenization mechanism where all non alphanumeric characters are removed and tokens are obtained by splitting on spaces.

Soft-Cosine. Soft-Cosine similarity was used in the winning approach for SemEval 2017 and is a successful technique for direct similarity computation. We use 100 dimensional CBOW word2vec representations trained on the titles of all questions in the database and follow (Charlet and Damnati, 2017) for the Soft-Cosine implementation. We consider this model as a direct similarity baseline.

SIF Embeddings. This is a popular and successful baseline for constructing sentence embeddings from individual token embeddings. For word embeddings we use the same vectors as described in Soft-Cosine and follow the implementation described in (Arora et al., 2017). We use cosine similarity over the representations to compute question similarity.

Universal Sentence Encoder. As described in (Cer et al., 2018) Universal Sentence Encoder (USE) is state of the art approach for universal text representations that can be effectively leveraged for many downstream tasks including similarity. We used the code released by the authors [4] to obtain question representations and use inner product for similarity.

5.2.3 Model Training

All models were implemented with *Keras* [5] using *TensorFlow* [6] backend. Both the reference and query log models, which share almost identical architectures, used 100 dimensional word embeddings as in the baselines and further tuned by each model while training. Both bidirectional LSTMs had 128 dimensional hidden states and the two fully connected layers had 64 dimensional hidden states and a RELU activation function. Between the two fully connected layers we applied dropout with a rate of 0.2. The question titles were truncated to 30 tokens and padded where needed before training. The model was trained using Adam optimization with initial learning rate of 0.001 and $\beta 1$ and $\beta 2$ are 0.9 and 0.999 respectively. Both models used batch size of 256 and the reference

[4] https://tfhub.dev/google/universal-sentence-encoder/2
[5] https://keras.io
[6] https://www.tensorflow.org

model was trained for 12 epochs while the query log model was trained for 70 epochs. Hyper-parameters were tuned using 10% validation set on the original supervised task for each model.

For the multi-task method we use the same word embeddings described in the baselines. due to the dot product operation we use 100 dimensional fully connected layers instead of the regular 64 used in the rest of the models. We minimize a weighted average $\alpha * query\_loss + (1-\alpha) * cite\_loss$, where $query\_loss$ is the mean of the squared dot products between query and question representations and $cite\_loss$ is the cross entropy of the reference predictions. α is set to 0.2 using a validation set on the original tasks. In the iterative method we use the same network setting as the individual models, with 8 iteration each, starting with an epoch with the query log target followed by an epoch with the reference target.

The models that exclusively use the details section are trained with the exact same settings as those using the titles. Since the details section is usually longer, the texts in this case are truncated to 100 tokens which resulted in truncation of 3% of the questions. Training a separate word embeddings for the details did not yield performance increase, and they are initialized with the embeddings learned from the titles and fine tuned during training. In the joint title-details models the sub-component parameters are identical to the networks described for individual models, where the fully connected layer after concatenation has 64 dimension hidden state and a dropout with 0.2 rate is performed between the concatenated vectors and the fully connected layer. For hierarchical attention networks we use a sentence tokenizer from nltk [7]. We follow the paper implementation for HAN, when we change the word embedding dimensions and the optimization scheme to the same as described for the titles.

5.3 Results

5.3.1 Title Methods

Figure 4 shows the similarity accuracy of methods based on exclusively the titles. The evaluation clearly shows that the supervised models significantly outperform all baselines. While on the fine-grained dataset the supervised models present almost equal performance on the coarse dataset the embeddings coming from the hidden layer of the reference model are significantly better than the rest. We explain the fact that the unsupervised models perform poorly by the unique semantics and vocabulary present in the tax domain. The fact that USE, being a fully transfer model, underperforms SIF which has the advantage of leveraging domain specific word vectors further supports this explanation. As expected, the coarse dataset is much easier for classification, with the accuracy of the best performing model on the fine-grained dataset is on par with the lower performing model on the coarse dataset.

[7]https://www.nltk.org/

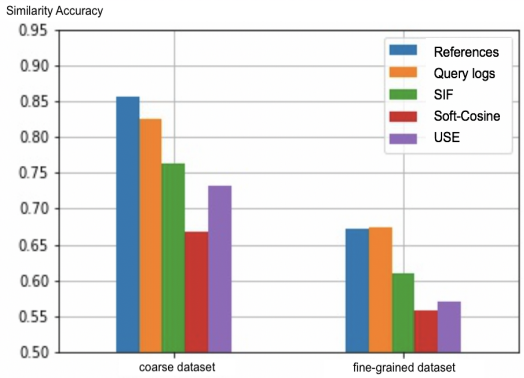

Figure 4: Comparison of title based embeddings on similarity prediction.

5.3.2 Combination Methods

Figure 5 presents the performance of the different combination methods mentioned in 4.3. We show the base two models and compare them to three combination methods. While not dramatic, there is significant improvement of the combined models on the fine-grained dataset. While the multi-task approach seems not to be effective, both concatenation, averaging and especially iterative training outperform each of the stand alone models on the fine-grained dataset with no performance loss on the coarse dataset. We address the poor performance of the multi-task approach to the low amount of questions having both labels. The success of the iterative model confirms our assumptions on the advantage of combining both supervised signals in one model.

5.3.3 Models with Details Section

When we combine the details section into the model, the results are less decisive with several models showing similar performance. The full results are listed in Table 4. The attempt to jointly learn the references from both title and details did not yield significant improvement over the title only representations (the results for the iterative and query log models are not listed but are inferior to other models as well). The addition of a more complex model for details, in the form of HAN, yielded very marginal im-

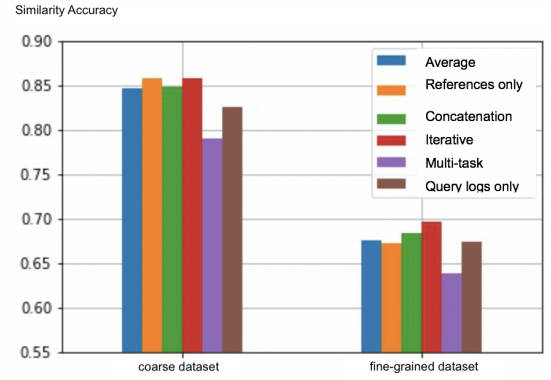

Figure 5: Similarity accuracy of combination methods of reference and query-log models.

provements. We address the overall disappointing results of the joint learning to the fact that the question representations were not significantly different from those learned with the titles exclusively. This is supported by the similar performance of the joint model to the models reported in Figure 4. The joint models also only marginally improved performance on the original supervised tasks. We leave the improvement of joint models for future work. We therefore focus on unsupervised combinations of separately learned representations for each section. In most cases, averaging and concatenation behaved very similar for all approaches and we report results for concatenation. On the fine-grained dataset, concatenation of representation learned in a supervised way were the best performing. Specifically the model in which both titles and details were separately iteratively trained on both references and query logs was the most successful one. Despite showing quite poor performance on titles only, Universal Sentence Encoder proved to be very useful in encoding the details and was only slightly inferior on the fine-grained dataset and the best performing model on the coarse dataset. We attribute this to the model's ability to model complex textual data which is present in the details section. As expected, SIF embedding (and moreover Soft-Cosine) struggled to model long texts in an effective way.

6 CONCLUSIONS AND FUTURE WORK

In this paper we presented methods for measuring similarity in CQA sites. We showed that learning the representation by using supervised text classification on proxy variables leads to a significant improvement over state of the art text embedding models. Moreover, when similarity was calculated from the titles

Table 4: Comparison of similarity accuracy of models which use both titles and details.

Model	**fine-grained dataset**	**coarse dataset**
Concatenation of references models	.7793	.9199
Concatenation of query log models	.7291	.8688
Concatenation of iteratively trained models	**.7964**	.9131
Joint learning on references	.6797	.8356
Joint learning on references with HAN	.6797	.8390
Concatenation of USE representations	.7640	.9293
Average of USE representations	.7606	**.9386**
Concatenation of SIF representations	.7027	.8475

only the advantage of our methods was even bigger. We also compared few methods to combine multiple supervised models and obtained that its superior to each model separately.

For future work we intend to explore several directions. First, the TTLC data contains relatively large number of high quality internal question references which made the reference prediction model very successful. However, the replies and reference mechanism contain additional structure that can be used, such as multiple references, the textual context of the reference and more. We expect that deeper analysis of all these would provide more training data in TTLC, as well as make our method more applicable to other CQAs.

Contrary to the high-quality reference data that we had, the query click data was relatively small and lacked a lot of important features. For example, the page dwell time could definitely improve our algorithm. The gap in quality between data sources also emphasizes the need for a good method to combine different sources of information with different level of confidence. Of course pure statistical methods like parameter tuning using cross-validation could be used, however since cross-validation is again relying on labeled data, it would be interesting to come with a near-optimal solution for which it is not required.

Finally, we are aware that NLP methods for text modeling are improving rapidly. In our experiments, Universal Sentence Encoder was highly accurate for representing the details section, probably because of its ability to capture the semantics of longer and more complicated texts. Lately, fine tuning pretrained

transformer based language models has achieved state of the art results on many NLP tasks. Incorporating the weak signals with those methods may further increase performance.

REFERENCES

Arora, S., Liang, Y., and Ma, T. (2017). A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017*.

Baeza-Yates, R. and Tiberi, A. (2007). Extracting semantic relations from query logs. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 76–85.

Bogdanova, D., dos Santos, C., Barbosa, L., and Zadrozny, B. (2015). Detecting semantically equivalent questions in online user forums. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 123–131. Association for Computational Linguistics.

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., and Specia, L. (2017). Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14. Association for Computational Linguistics.

Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Strope, B., and Kurzweil, R. (2018). Universal sentence encoder for english. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174. Association for Computational Linguistics.

Charlet, D. and Damnati, G. (2017). Simbow at semeval-2017 task 3: Soft-cosine semantic similarity between questions for community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 315–319. Association for Computational Linguistics.

Chopra, S., Hadsell, R., and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Craswell, N. and Szummer, M. (2007). Random walks on the click graph. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 239–246.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Ein Dor, L., Mass, Y., Halfon, A., Venezian, E., Shnayderman, I., Aharonov, R., and Slonim, N. (2018). Learning thematic similarity metric from article sections using triplet networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 49–54. Association for Computational Linguistics.

Figueroa, A. and Neumann, G. (2013). Learning to rank effective paraphrases from query logs for community question answering. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, AAAI'13, pages 1099–1105.

Filice, S., Da San Martino, G., and Moschitti, A. (2017). Kelp at semeval-2017 task 3: Learning pairwise patterns in community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 326–333. Association for Computational Linguistics.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Hoffer, E. and Ailon, N. (2015). Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer.

Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., and Heck, L. (2013). Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pages 2333–2338, New York, NY, USA. ACM.

İrsoy, O. and Cardie, C. (2014). Opinion mining with deep recurrent neural networks. In *EMNLP*, pages 720–728.

Jeh, G. and Widom, J. (2002). Simrank: A measure of structural-context similarity. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02.

Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R. S., Torralba, A., Urtasun, R., and Fidler, S. (2015). Skip-thought vectors. *arXiv preprint arXiv:1506.06726*.

Logeswaran, L. and Lee, H. (2018). An efficient framework for learning sentence representations. In *International Conference on Learning Representations ICLR 2018*.

Ma, H., Yang, H., King, I., and Lyu, M. R. (2008). Learning latent semantic relations from clickthrough data for query suggestion. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 709–718, New York, NY, USA.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words

and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Mueller, J. and Thyagarajan, A. (2016). Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2786–2792.

Nakov, P., Hoogeveen, D., Màrquez, L., Moschitti, A., Mubarak, H., Baldwin, T., and Verspoor, K. (2017). Semeval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 27–48.

Nassif, H., Mohtarami, M., and Glass, J. (2016). Learning semantic relatedness in community question answering using neural models. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 137–147. Association for Computational Linguistics.

Neculoiu, P., Versteegh, M., and Rotaru, M. (2016). Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 148–157. Association for Computational Linguistics.

Pagliardini, M., Gupta, P., and Jaggi, M. (2018). Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540.

Poblete, B. and Baeza-Yates, R. (2008). Query-sets: Using implicit feedback and query patterns to organize web documents. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 41–50.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098.

Schuster, M. and Paliwal, K. (1997). Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681.

Shao, Y. (2017). Hcti at semeval-2017 task 1: Use convolutional neural network to evaluate semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 130–133. Association for Computational Linguistics.

Srba, I. and Bieliková, M. (2016). A comprehensive survey and classification of approaches for community question answering. *TWEB*, 10:18:1–18:63.

Wu, H., Wu, W., Zhou, M., Chen, E., Duan, L., and Shum, H.-Y. (2014). Improving search relevance for short queries in community question answering. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, pages 43–52, New York, NY, USA.

Wu, W., Li, H., and Xu, J. (2013). Learning query and document similarities from click-through bipartite graph with metadata. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 687–696.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489. Association for Computational Linguistics.

Zhang, W. E., Sheng, Q. Z., Lau, J. H., and Abebe, E. (2017). Detecting duplicate posts in programming qa communities via latent semantics and association rules. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 1221–1229.