GROUP SPARSE CNNS FOR QUESTION SENTENCE CLASSIFICATION WITH ANSWER SETS

Mingbo Ma & Liang Huang
Department of EECS
Oregon State University
Corvallis, OR 97331, USA
{mam,liang.huang}@oregonstate.edu

Bing Xiang & Bowen Zhou
IBM Watson Group
T. J. Watson Research Center, IBM
Yorktown Heights, NY 10598, USA
{bingxia,zhou}@us.ibm.com

ABSTRACT

Classifying question sentences into their corresponding categories is an important task with wide applications, for example in many websites' FAQ sections. However, traditional question classification techniques do not fully utilize the well-prepared answer data which has great potential for improving question representation and could lead to better classification performance. In order to encode answer information into question representation, we first introduce novel group sparse autoencoders which could utilize the group information in the answer set to refine question representation. We then propose a new group sparse convolutional neural network which could naturally learn the question representation with respect to their corresponding answers by implanting the group sparse autoencoders into the traditional convolutional neural network. The proposed model show significant improvements over strong baselines on four datasets.

1 INTRODUCTION

Question classification has applications in question answering (QA), dialog systems, etc., and has been increasingly popular in recent years. Most existing approaches to this problem simply use existing sentence modeling frameworks and treat questions as general sentences, without any special treatment. For example, several recent efforts employ Convolutional Neural Networks (CNNs) to achieve remarkably strong performance in the TREC question classification task as well as other sentence classification tasks such as sentiment analysis (Kim, 2014; Kalchbrenner et al., 2014; Ma et al., 2015).

We argue, however, that the general sentence modeling frameworks neglect several unique properties in question classification not found in other sentence classification tasks (such as sentimental classification or sarcasm detection), which we detail below:

- The categories for most sentence classification tasks are flat and coarse (notable exceptions such as the Reuters Corpus RCV1 (Lewis et al., 2004) notwithstanding), and in many cases, even binary (i.e. sarcasm detection). However, question sentences commonly belong to multiple categories, and these categories often have a hierarchical (tree or DAG) structure such as those from the New York State DMV FAQ section [1] in Fig. 1.
- Question sentences from different categories often share similar information or language patterns. This phenomenon becomes more obvious when categories are hierarchical. Fig. 2 shows one example of questions sharing similar information from different categories. This cross-category shared patterns are not only shown in questions but can also be found in answers corresponding to these questions.
- Another unique characteristic for question classification is the well prepared answer set with detailed descriptions or instructions for each corresponding question category. These answer sets generally cover a broader range of vocabulary (than the questions themselves) and carry more distinctive semantic meanings for each class. We believe there is great

[1] http://nysdmv.custhelp.com/app/home

1: Driver License/Permit/Non-Driver ID
a: *Apply for original* (49 questions)
b: *Renew or replace* (24 questions)
...
2: Vehicle Registrations and Insurance
a: *Buy, sell, or transfer a vehicle* (22 questions)
b: *Registration and title requirements* (42 questions)
...
3: Driving Record / Tickets / Points
...

Figure 1: Examples from the NYDMV FAQ section. There are 8 top-level categories, 47 sub-categories, and 537 questions (388 *unique* questions; many questions fall into multiple categories).

Category: Finance
Q: How to get a personal loan from the bank?
Category: Education
Q: What are the steps for applying for student loan?

Figure 2: Examples of questions from two different categories. These questions ask for the similar problem even if they are in different classes. Their answers also contain similar information.

potential to enhance the representation of questions with extra information from corresponding answer sets.

To exploit the hierarchical and overlapping structures in question categories and extra information from answer sets, we consider dictionary learning (Aharon et al., 2005; Roth & Black, 2005; Lee et al., 2007; Candè & Wakin, 2008; Kreutz-Delgado et al., 2003; Rubinstein et al., 2010) which is one common approach for representing samples from a vast, correlated groups with external information. This learning procedure first builds a dictionary with a series of grouped bases. These bases can be initialized randomly or from external data (from the answer set in our case) and optimized during training through Sparse Group Lasso (SGL) (Simon et al., 2013). There are many promising improvements which have been achieved recently by this grouped-dictionary learning-based methods (Zhao et al., 2016; Rao et al., 2016). We also showcase some preliminary experiments in Section 6 for question classification with SGL, and the performance is indeed extraordinary compared with baselines but still lose to the CNNs-based method. Considering the unique advantages from the SGL-based model and the CNNs-based model, we believe that performance of question classification will have another boost if we could put SGL-based and CNNs-based model within the same end-to-end framework. This requires us to design a new neural-based model which behaves similarly with SGL.

Based on the above observations, we first propose a novel Group Sparse Autoencoders (GSA). The objective of GSA and SGL are very similar. The encoding matrix of GSA (like the dictionary in SGL) is grouped into different categories. The bases in different groups can be either initialized randomly or by the sentences in corresponding answer categories. Each question sentence will be reconstructed by some bases within a few groups. To the best of our knowledge, GSA is the first full neural network based model with group sparse constraints. GSA has can be either linear or nonlinear encoding or decoding while SGL is restrained to be linear. In order to incorporate both advantages from GSA and CNNs, we then propose a new Group Sparse Convolutional Neural Networks (GSCNNs) by implanting the GSA into CNNs between the convolutional layer and the classification layer. GSCNNs are jointly trained end-to-end neural-based framework for getting question representations with group sparse constraint from both answer and question sets. Experiments show significant improvements over strong baselines on four datasets.

2 PRELIMINARIES: SPARSE AUTOENCODERS

We first review the basic autoencoders and sparse autoencoders to establish the mathematical notations. Then we propose our new autoencoder with group sparse constraints in later section.

2.1 BASIC AUTOENCODERS

As introduced in (Bengio et al., 2007), autoencoder is an unsupervised neural network which could learn the hidden representations of input samples. An autoencoder takes an input instance $\mathbf{z} \in R^d$, and then maps it into a hidden space in the form of $h \in R^s$ through a deterministic mapping function $h = \Phi_\theta(\mathbf{z}) = \Phi(W\mathbf{z} + b)$, where $\theta = \{W, b\}$. W is a $d \times s$ projection matrix and b is the bias term. The projection function can be linear or non-linear function such as sigmoid. This projection process often can be recognized as encoding process. The encoded hidden representation is then mapped back to the original input space to reconstruct a vector $\hat{\mathbf{z}} \in R^d$ with function $\hat{\mathbf{z}} = \Phi_{\theta'}(h) = \Phi(W'h + c)$ with $\theta' = \{W', c\}$. The reverse projection matrix W' may optionally be constrained by $W' = W^T$. This reverse operation can be recognized as a decoding process which tries to reconstruct a new vector $\mathbf{z}$ such that the difference between $\hat{\mathbf{z}}$ and $\mathbf{z}$ are as small as possible by minimizing the average reconstruction error:

$$J(W, b, c) = \underset{W,b,c}{\mathbf{argmin}} \frac{1}{n} \sum_{i=1}^{n} L(\mathbf{z}^{(i)}, \hat{\mathbf{z}}^{(i)}) = \underset{W,b,c}{\mathbf{argmin}} \frac{1}{n} \sum_{i=1}^{n} L\Big(\mathbf{z}^{(i)}, \Phi_{W^T,c}(\Phi_{W,b}(\mathbf{z}^{(i)}))\Big) \tag{1}$$

where L is a loss function such as minimum square error $L(\mathbf{z}, \hat{\mathbf{z}}) = \|\mathbf{z} - \hat{\mathbf{z}}\|^2$. Depending on the applications, this loss function also can be defined in form of computing the reconstruction cross-entropy between $\mathbf{z}$ and $\hat{\mathbf{z}}$:

$$L_C(\mathbf{z}, \hat{\mathbf{z}}) = -\sum_{k=1}^{d} (z_k \log \hat{z}_k + (1 - z_k) \log(1 - \hat{z}_k))$$

When the dimensionality of the hidden space s is smaller than the dimensionality of the input space d. The network is forced to learn a compressed representation of the input. If there is structure or feature correlation in the data, the linear autoencoders often ends up learning a low-dimensional representation like PCA. Most of the time, autoencoders learns a compressed representation when the number of hidden units s being small. However, when the number of hidden units becomes larger than the dimensionality of input space, there are still some interesting structure that can be discovered by imposing other constraints on the network. The following discussed sparse constraints is one of them.

2.2 SPARSE AUTOENCODERS

Sparse autoencoders (Ng, 2011; Makhzani & Frey, 2014) shows interesting results of getting visualization of the hidden layers. Recall that h_j^i represents the activations of j^{th} hidden unit for a given specific input $\mathbf{z}_i$. Then the average activation of hidden unit j (average over the training batch) can be defined as:

$$\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^{m} h_j^i \tag{2}$$

where m is the number of samples in training batch. The goal of sparse autoencoders is to enforce the constraint:

$$\hat{\rho}_j = \rho \tag{3}$$

where ρ is the sparsity parameter which controls how sparse you want the hidden representation to be. Typically ρ is set to be a small value close to zero. In order to satisfy this constraint, the activations of hidden layer must mostly be close to 0.

In order to achieve the above objective, there will be an extra penalty term in our optimization function which tries to reconstruct the original input with as few hidden layer activations as possible. The most commonly used penalty term (Ng, 2011) is as follows:

$$\sum_{j=1}^{s} KL(\rho || \hat{\rho}_j) = \sum_{j=1}^{s} \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j} \tag{4}$$

where s is the number of units in hidden layer, and j is the index of the hidden unit. This penalty term is based on KL divergence which measures the difference between two different distributions.

Then our new objective of the sparse autoencoders is defined as follows:

$$J_{sparse}(W, b, c) = J(W, b, c) + \alpha \sum_{j=1}^{s} KL(\rho||\hat{\rho}_j) \tag{5}$$

where $J(W, b, c)$ is defined in Eq. 1, and α controls the weights of the sparsity penalty term. Note that the term $\hat{\rho}_j$ is implicitly controlled by W, b and c. This is one of the difference between sparse autoencoders and sparse coding which will be discussed in details in Section 6.

3 GROUP SPARSE AUTOENCODERS

As described above, sparse autoencoder has similar objective with sparse coding which tries to find sparse representations for input samples. Inspired by the motivations from group sparse lasso (Yuan & Lin, 2006) and sparse group lasso (Simon et al., 2013), we propose a novel Group Sparse Autoencoders (GSA)in this paper.

Different from sparse autoencoders, in our GSA, the weight matrix is categorized into different groups. For a given input, GSA reconstructs the input signal with the activations from only a few groups. Similar to the average activation defined in Eq. 2 for sparse autoencoders, in GSA, we define each grouped average activation for the hidden layer as follows:

$$\hat{\eta}_p = \frac{1}{mg} \sum_{i=1}^{m} \sum_{l=1}^{g} ||h_{p,l}^{i}||_2 \tag{6}$$

where g represents the number of samples in each group, and m represents the number of samples in training batch. $\hat{\eta}_j$ first sums up all the activations within p^{th} group, then computes the average p^{th} group respond across different samples' hidden activations.

Similar with Eq.4, we also use KL divergence to measure the difference between estimated intra-group activation and goal group sparsity as follows:

$$\sum_{p=1}^{G} KL(\eta||\hat{\eta}_p) = \eta \log \frac{\eta}{\hat{\eta}_p} + (1 - \eta) \log \frac{1 - \eta}{1 - \hat{\eta}_p} \tag{7}$$

where G is the number of groups. When we only need inter-group constraints, the loss function of autoencoders can be defined as follows:

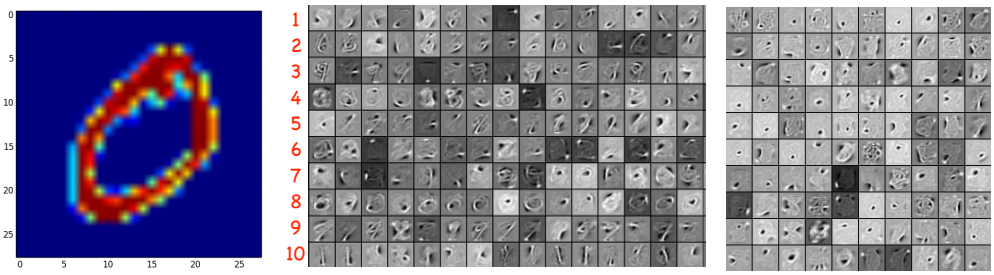
$$J_{gs}(W, b, c) = J(W, b, c) + \beta \sum_{l=1}^{g} KL(\eta||\hat{\eta}_p) \tag{8}$$

In some certain cases, inter- and intra- group sparsity are preferred and the same time. Then objective can be defined as follows:

$$\begin{aligned} J_{gs}(W, b, c) = J(W, b, c) &+ \alpha \sum_{j=1}^{s} KL(\rho||\hat{\rho}_j) \\ &+ \beta \sum_{p=1}^{G} KL(\eta||\hat{\eta}_p) \end{aligned} \tag{9}$$

Inter-group sparse autoencoders defined in Eq. 8 has similar functionality with group sparse lasso in (Yuan & Lin, 2006). Inter- and intra- group sparse autoencoders which defined in Eq. 9 behaves similarly to sparse group lasso in (Simon et al., 2013). Different from the sparse coding approach, the encoding and decoding process could be nonlinear while sparse coding is always linear.

Similar to sparse coding approach, the projection matrix in GSA works like a dictionary which includes all the necessary bases for reconstructing the input signal with the activations in the hidden layer. Different initialization methods for projection matrix are described in Section 5.



(a) (b) (c)

Figure 3: The input figure with hand written digit 0 is shown in (a). Figure (b) is the visualization of projection matrix W. Different rows represent different groups of W in Eq. 9. For each group, we only show the first 15 (out of 50) bases. The red numbers on the left side of (b) are the index of different groups(10 groups in total). Figure (c) is the projection matrix visualization from a basic autoencoders.

3.1 Visualization for Group Sparse Autoencoders

In order to have a better understanding of how the GSA behaves, We use MNIST dataset for visualizing the internal parameters of GSA. We visualize the projection matrix in Fig. 3 and the corresponding hidden activation in Fig. 4.

In our experiments, we use $10,000$ samples for training. We set the size of hidden layer as 500 with 10 different groups for GSA. We set the intra-group sparsity ρ equal to 0.3 and inter-group sparsity η equal to 0.2. α and β are equal to 1. On the other hand, we also train the same $10,000$ examples on basic autoencoders with random noise added to the input signal (denoising autoencoders (Vincent et al., 2008)) for better hidden information extraction. We add the same 30% random noise into both models. Note that the group size of this experiments does not have to be set to 10. Since this is the image dataset with digit numbers, we may use fewer groups to train GSA.

In Fig. 3(b), we could find similar patterns within each group. For example, the 8^{th} group in Fig. 3(b) has different forms of digit 0, and 9^{th} group includes different forms of digit 7. However, it is difficult to tell any meaningful patterns from the projection matrix of basic autoencoders in Fig. 3(c).

Fig. 4 shows the hidden activations respect to the input image in Fig. 3(a). From the results, we can tell that most of the activations of hidden layer are in group 1, 2, 6 and 8. And the 8^{th} group has the most significant activations. When we refer this activations to the projection matrix visualization in Fig. 3(b). These results are reasonable since the 8^{th} row has the most similar patterns of digit 0.

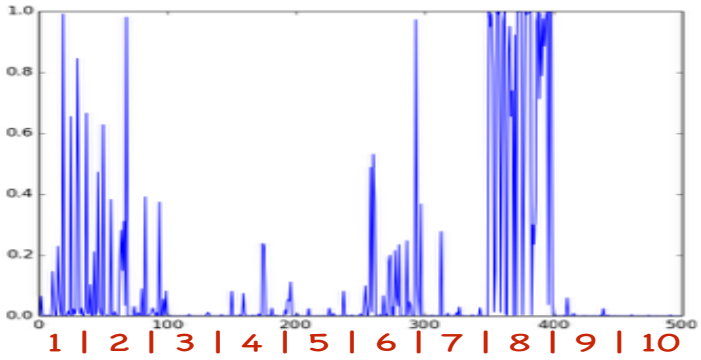


Figure 4: The hidden activations h respect to the input image in Fig. 3(a). The red numbers corresponds to the index in Fig. 3(b). These activations come from 10 different groups. The group size here is 50.

GSA could be directly applied to small image data (i.e. MINIST dataset) for pre-training. However, in the tasks which prefer dense, semantic representation (i.e. sentence classification), we still need CNNs to learn the

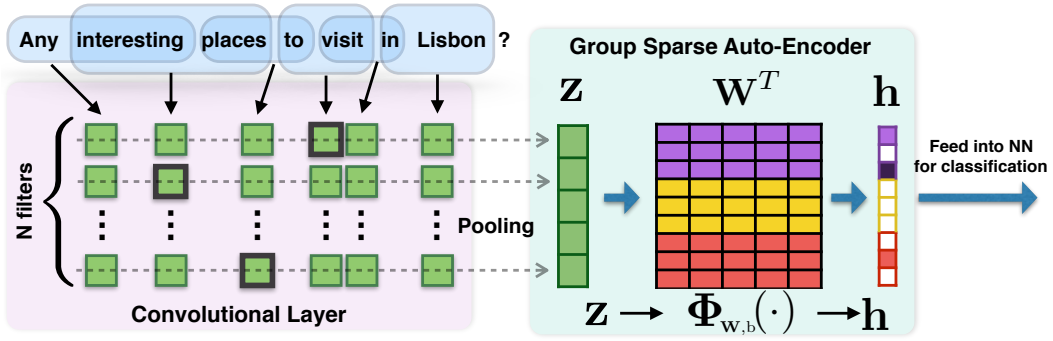


Figure 5: Framework used in our model. We add extra encoding layer in CNNs. Sentence representation after convolutional layer is denoted as $\mathbf{z}$, and W is the projection matrix (functions as dictionary) in Eq. 9. Hidden group sparse representation for question sentence is denoted as $\mathbf{h}$. Different colors in projection matrix represent different groups. We show W^T instead of W in the figure for better visualization purpose. The darker color in $\mathbf{h}$ means larger value and white means zero.

sentence representation automatically. In this scenario, in order to incorporate both advantages from GSA and CNNs, we propose Group Sparse Convolutional Neural Networks in the following section.

4 Group Sparse Convolutional Neural Nets

Convolutional neural networks (CNNs) were first proposed by (LeCun et al., 1995) in computer vision. For a given image, CNNs apply convolution kernels on a series of continuous areas on images. This concept was first adapted to NLP by (Collobert et al., 2011). Recently, many CNNs-based techniques achieve great successes in sentence modeling and classification (Kim, 2014; Kalchbrenner et al., 2014; Ma et al., 2015). For simplicity, we use the sequential CNNs (Kim, 2014) as our baseline.

Following sequential CNNs, one dimensional convolution operates the convolution kernel in sequential order in Eq. 10, where $\mathbf{x}_i \in \mathbb{R}^e$ represents the e dimensional word representation for the i-th word in the sentence, and $\oplus$ is the concatenation operator. Therefore $\mathbf{x}_{i,j}$ refers to concatenated word vector from the i-th word to the $(i+j)$-th word in sentence:

$$\mathbf{x}_{i,j} = \mathbf{x}_i \oplus \mathbf{x}_{i+1} \oplus \cdots \oplus \mathbf{x}_{i+j} \tag{10}$$

A convolution operates a filter $\mathbf{w} \in \mathbb{R}^{n \times e}$ to a window of n words $\mathbf{x}_{i,i+n}$ with bias term b' described in Eq. 11 to produce a new feature.

$$a_i = \sigma(\mathbf{w} \cdot \mathbf{x}_{i,i+n} + b') \tag{11}$$

where σ is a non-linear activation function such as rectified linear unit (ReLu) or sigmoid function. The filter $\mathbf{w}$ is applied to each word in the sentence, generating the feature map $\mathbf{a} \in \mathbb{R}^L$:

$$\mathbf{a} = [a_1, a_2, \cdots, a_L] \tag{12}$$

where L is the length of the sentence.

The convolution described in Eq. 11 can be regarded as feature detection: more similar patterns will return higher activation. In sequential CNNs, max-over-time pooling (Collobert et al., 2011; Kim, 2014) operates over the feature map to get the maximum activation $\hat{a} = \max\{\mathbf{a}\}$ representing the entire feature map. The idea is to detect the strongest activation over time. This pooling strategy also naturally deals with sentence length variations.

In order to capture different aspects of patterns, CNNs usually randomly initialize a set of filters with different sizes and values. Each filter will generate a feature as described above. To take all the features generated by N different filters into count, we use $\mathbf{z} = [\hat{a_1}, \cdots, \hat{a_N}]$ as the final representation.

In conventional CNNs, $\mathbf{z}$ will be directly fed into classifiers after the sentence representation is obtained, e.g. fully connected neural networks in (Kim, 2014). There is no easy way for CNNs to explore the possible hidden representations with interesting underlaying structures.

In order to obtains the hidden representations for each sentence representation, we proposed a Group Sparse Convolutional Neural Networks (GSCNNs) by placing one extra layer between convolutional layer and classification layer. This extra layer is trying to mimic the functionality of GSA that we introduced in Section 2.

Our proposed framework is shown in Fig. 5. The convolutional layer show in Fig. 5 follows the traditional convolution process which is described previously. After the convolutional layer, we get $\mathbf{z}$ which is the feature map for each sentence. The feature maps $\mathbf{z}$ is treated as the feature representation for each sentence. In stead of directly feeding $\mathbf{z}$ into a fully connected neural network for classification, we enforce the group sparse constraint on z like the group sparse constraint we have on h in Eq. 9. Then, we use the hidden representation h in Eq. 9 as new sentence representation. The last step is feeding the hidden representation h into fully connected neural network for classification. The parameters W, b, and c in Eq. 9 will also be fine tunned during the last step.

In order to improve the robustness of the hidden representation and prevent it from simply learning the identity, we follow the idea of decisioning autoencoders (Vincent et al., 2008) to add random noise (10% in our experiments) into $\mathbf{z}$. The training process of our model is similar to the training process in stack autoencoders (Bengio et al., 2007).

In order to prevent the co-adaptation of the hidden unites, we employ random dropout on penultimate layer (Hinton et al., 2014). We set the drop out rate as 0.5 and learning rate as 0.95 by default. In our experiments, training is done through stochastic gradient descent over shuffled mini-batches with the Adadelta update rule (Zeiler, 2012). All the settings of the CNNs are the same as the settings in (Kim, 2014).

5 EXPERIMENTS

Since there has been little effort to use answer sets in question classification, we did not find any well-fitted datasets which are publicly available. We collected two datasets and use other two well-known datasets in our experiments. The statistics of these datasets is summarized in Table 1. The descriptions of each dataset are as follows:

- **TREC** The TREC dataset[2] is a factoid question classification dataset. The task is to classify each question into one of the 6 different question types (Li & Roth, 2002). The reason we include this factoid questions dataset is to show the effectiveness of the proposed method in an frequently used dataset even there is no categorized answer sets available.
- **Insurance** This is a private dataset which we collected from a car insurance company's website. Each question is classified into the 319 possible classes with corresponding answer data. All questions which belongs to the same category share the same answers. All answers are generated manually. Most questions have multiple assigned labels.
- **DMV dataset** We collected this dataset from New York State DMV's FAQ website. We will make this data publicly available in the future.
- **Yahoo Ans** The Yahoo! Answers dataset (Fleming et al., 2012; Shah & Pomerantz, 2010) is a publicly available dataset.[3] There are more than 4 million questions with answers. For simplicity reasons, we only randomly sample 8,871 questions from the complete dataset. There are 27 top level categories across different domains. To make our task more realistic and challenging, we test the proposed model with respect to the subcategories and there are 678 classes.

Datasets	C_t	C_s	N_{data}	N_{test}	N_{ans}	Multi-label ?
TREC	6	50	5952	500	-	No
Insurance	-	319	1580	303	2176	Yes
DMV	8	47	388	50	2859	Yes
Yahoo Ans	27	678	8871	3027	10365	No

Table 1: Summary of dataset statistics. C_t represent the number of top categories, and C_s represents the number of sub-category. Note we only do top level classification on TREC. N_{data} is dataset size. N_{test} is the size for test set. N_{ans} is the size of answer set.

[2] http://cogcomp.cs.illinois.edu/Data/QA/QC/

[3] http://webscope.sandbox.yahoo.com/catalog.php?datatype=l

	TREC	Insurance	DMV	Yahoo dataset		
				sub	top	unseen
CNNs	93.6	51.2	60	20.8	53.9	47
W_R	93.8	53.5	62	21.8	54.5	48
W_Q	**94.2**	53.8	64	22.1	54.1	48
W_A	-	**55.4**	**66**	**22.2**	**55.8**	**53**

Table 2: Experiments with four datasets. Baseline is from sequential CNNs. W_R means the projection matrix is random initialized. W_Q represents the projection matrix is initialized by clustering the question sentences. W_A represents the performance of the model whose projection matrix is initialized by answer set. There are three different settings for Yahoo dataset: classification on subcategory, classification on top level category and classification on unseen sub-labels.

We only compare our model's performance with CNNs for two following reasons: we consider our "group sparse" as a modification to the general CNNs for grouped feature selection. This idea is "orthogonal" to any other CNNs-based models and can be easily applied to them; another reason is, as discussed in Sec. 1, we did not find any other model which can be used for comparison in soloving question classification task with answer sets.

The datasets we use in the experiments require the label information for both questions and answers. Besides that, similar with websites' FAQ section, all the questions which belong to the same category share the same answer sets. Among the above the four datasets, only the Insurance and DMV datasets are well-fitted for our model. The questions which fall into the same category have different answers in Yahoo dataset.

Different ways of initializing the projection matrix in Eq. 9 can be summarized as the followings:

- **Random Initialization**: when there is no answer corpus available, we first random initialize N vectors (usually $N \gg s$) to represent the representation from answer set. Then we cluster these N vectors into G categories with g centroids for each category. These centroids from different categories will be the initialized bases for projection matrix W. This projection matrix will be optimized during training.
- **Initialization from Questions**: instead of using random initialized vectors, we could also use question sentences for initializing the projection matrix when answer set is not available. We need to pre-train the sentence with CNNs to get the sentence representation. We then select top G largest categories in terms of number of question sentences. Then we get g centroids from each category by k-means. We concatenate these $G \times g$ vectors group after group to form the projection matrix. We need to pre-train the sentence with CNNs to get the sentence representation.
- **Initialization from Answers**: This is the most ideal case. We follow the same procedure from above. The only difference is that we then treat the answer sentence as question sentence to pre-train the CNNs to get answer sentence representation.

Note that projection matrix will be updated during training for better classification performance.

In the cases of single-label classification tasks (TREC and Yahoo dataset), we set the last layer as softmax-layer which tries to get one unique peaky choice across all other labels. But in the cases for multi-label classification (Insurance and DMV dataset), we replace the softmax-layer in CNNs with sigmoid-layer since sigmoid layer predicts each category independently while softmax function has an exclusive property which allows cross influence between categories.

All the experiments results are summarized in Table 2. TREC dataset is factoid question type classification. We include this experiments to show our performance on a frequently used dataset. Proposed method improves marginally over baseline because the sentences are too short in TREC dataset. For Insurance and DMV dataset, the improvement is significant.

In the experiments with Yahoo dataset, the improvement is not as signification as Insurance and DMV. One reason for this is the questions in Yahoo dataset are usually too short, sometime only have 2 to 3 words. When the sentences become shorter, the group information become harder to encode. Another reason is that the questions in Yahoo dataset are always single labeled, and can not fully utilize the benefits of group sparse properties. Yahoo-top shows the results of top categories classification results. We map the subcategories back to the top categories and get the results in Table 2.

Besides the conventional classification tasks, we also test our proposed model on unseen-label experiments. In this experiments, there are a few sub-category labels that are not included in training process. However, we still hope that our model could correctly classify these unseen sub-category label into correct parent category based on the model's sub-category estimation. In the testing set of Yahoo dataset, we randomly add 100 questions

k-NN based Model	vanilla k-NN	31.2
	k-NN + SGL	**32.2**
SVM based Model	vanilla SVM	33.7
	SVM + SGL	**44.5**
CNNs based Model	vanilla CNNs	51.2

Table 3: Experiments for two baseline model, k-NN and SVM, for the Insurance dataset.

whose labels are unseen in training set. The classification results of Yahoo-unseen in Table 2 are obtained by mapping the subcategory classification results to top level category and check whether the true label's top category match with predicted label's parent category. The improvements are remarkable due to the group information encoding.

6 DISCUSSION

The idea of reforming signal to a sparse representation is first introduced in the domain of compressed sensing (Candè & Wakin, 2008) which achieves great success in signal compression, visualization and classification task. Especially when dictionary is well trained, the performance usually improves significantly, as shown in (Wang et al., 2010; Yang et al., 2009) for image classification tasks. In Table 3, we test the influence of Sparse Group Lasso (SGL) (Simon et al., 2013) with two baseline methods, k-Nearest Neighbor (k-NN) and SVM on the Insurance dataset. We use TF-IDF as feature representation for each question and answer sentence. We first select all the answer sentences from top 20 largest category and then find 10 centroids for each of these categories by k-Means. Then we have a dictionary with 200 centroids with 20 groups. We notice there is a great improvement of performance after we preprocess the original sentence representations with SGL before we use SVM. We further test the performance of CNNs on the same dataset, and CNNs outperforms SVM and k-NN even with SGL because of the well trained sentence representation through CNNs. However, for vanilla CNNs, it is not straightforward to embed SGL into the network and still get good representation for sentences since SGL will break the training error in backpropagation.

However, GSA is fully neural network based framework. Our proposed GSA has similar functionalities to SGL (Yuan & Lin, 2006; Simon et al., 2013), as it is shown in Fig. 3 and Fig. 4, but in different approach. Compared with sparse coding approaches which have intense optimizations on both dictionary and coding, GSA's optimization is based on simple backpropagation. GSA also can be easily placed into any neural network for joint training. Another advantage of GSA over sparse coding is that the projection function Φ in GSA can be linear or non-linear, while sparse coding always learns linear codings.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we first present a novel GSA framework which functions as dictionary learning and sparse coding models with inter- and intra- group sparse constraints. We also prove GSA's learning ability by visualizing the projection matrix and activations. We further propose a group sparse convolutional neural networks by embedding GSA into CNNs. We show that CNNs can benefit from GSA by learning more meaningful representation from dictionary.

REFERENCES

Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: Design of dictionaries for sparse representation. In *IN: PROCEEDINGS OF SPARS05*, pp. 9–12, 2005.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In B. Schölkopf, J.C. Platt, and T. Hoffman (eds.), *Advances in Neural Information Processing Systems 19*, pp. 153–160. MIT Press, 2007. URL http://papers.nips.cc/paper/3048-greedy-layer-wise-training-of-deep-networks.pdf.

Emmanuel J. Candè and Michael B. Wakin. An Introduction To Compressive Sampling. In *Signal Processing Magazine, IEEE*, volume 25, 2008. URL http://dx.doi.org/10.1109/msp.2007.914731.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. volume 12, pp. 2493–2537, 2011.

Simon Fleming, Dan Chalmers, and Ian Wakeman. A deniable and efficient question and answer service over ad hoc social networks. volume 15, pp. 296–331. Springer Netherlands, 2012. doi: 10.1007/s10791-012-9185-0. URL http://dx.doi.org/10.1007/s10791-012-9185-0.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *Journal of Machine Learning Research*, 15, 2014.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 655–665, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P14-1062.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D14-1181.

Kenneth Kreutz-Delgado, Joseph F. Murray, Bhaskar D. Rao, Kjersti Engan, Te-Won Lee, and Terrence J. Sejnowski. Dictionary learning algorithms for sparse representation. 2003.

Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Mller, E. Sckinger, P. Simard, and V. Vapnik. Comparison of learning algorithms for handwritten digit recognition. In *INTERNATIONAL CONFERENCE ON ARTIFICIAL NEURAL NETWORKS*, pp. 53–60, 1995.

Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *In NIPS*, pp. 801–808. NIPS, 2007.

David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397, 2004.

Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pp. 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1072228.1072378. URL http://dx.doi.org/10.3115/1072228.1072378.

Mingbo Ma, Liang Huang, Bing Xiang, and Bowen Zhou. Dependency-based convolutional neural networks for sentence embedding. In *Proceedings of ACL 2015*, 2015.

Alireza Makhzani and Brendan Frey. K-sparse autoencoders. In *International Conference on Learning Representations*. 2014.

Andrew Ng. Sparse autoencoder. 2011. URL https://web.stanford.edu/class/cs294a/sparseAutoencoder.pdf.

Nikhil Rao, Robert Nowak, Christopher Cox, and Timothy Rogers. Classification with the sparse group lasso. *IEEE Transactions on Signal Processing*, 64(2):448–463, 2016.

Stefan Roth and Michael J. Black. Fields of experts: A framework for learning image priors. In *In CVPR*, pp. 860–867, 2005.

R. Rubinstein, A. M. Bruckstein, and M. Elad. Dictionaries for sparse representation modeling. 2010.

Chirag Shah and Jefferey Pomerantz. Evaluating and predicting answer quality in community qa. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pp. 411–418, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0153-4. doi: 10.1145/1835449.1835518. URL http://doi.acm.org/10.1145/1835449.1835518.

Noah Simon, Jerome Friedman, Trevor Hastie, and Rob Tibshirani. A sparse-group lasso. 2013.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. pp. 1096–1103, 2008.

Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *IN: IEEE CONFERENCE ON COMPUTER VISION AND PATTERN CLASSIFICATOIN*, 2010.

Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *in IEEE Conference on Computer Vision and Pattern Recognition(CVPR*, 2009.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. volume 68, pp. 49–67, 2006.

Mattgew Zeiler. Adadelta: An adaptive learning rate method. *Unpublished manuscript*: http://arxiv.org/abs/1212.5701, 2012.

Yize Zhao, Matthias Chung, Brent A Johnson, Carlos S Moreno, and Qi Long. Hierarchical feature selection incorporating known and novel biological information: Identifying genomic features related to prostate cancer recurrence. *Journal of the American Statistical Association*, (just-accepted), 2016.