A Generalized Recurrent Neural Architecture for Text Classification with Multi-Task Learning

Honglun Zhang[1], **Liqiang Xiao**[1], **Yongkun Wang**[2], **Yaohui Jin**[1,2]

[1]State Key Lab of Advanced Optical Communication System and Network
[2]Network and Information Center
Shanghai Jiao Tong University
{ykw}@sjtu.edu.cn

Abstract

Multi-task learning leverages potential correlations among related tasks to extract common features and yield performance gains. However, most previous works only consider simple or weak interactions, thereby failing to model complex correlations among three or more tasks. In this paper, we propose a multi-task learning architecture with four types of recurrent neural layers to fuse information across multiple related tasks. The architecture is structurally flexible and considers various interactions among tasks, which can be regarded as a generalized case of many previous works. Extensive experiments on five benchmark datasets for text classification show that our model can significantly improve performances of related tasks with additional information from others.

1 Introduction

Neural network based models have been widely exploited with the prosperities of *Deep Learning* [Bengio *et al.*, 2013] and achieved inspiring performances on many NLP tasks, such as text classification [Chen *et al.*, 2015; Liu *et al.*, 2015a], semantic matching [Liu *et al.*, 2016d; 2016a] and machine translation [Sutskever *et al.*, 2014]. These models are robust at feature engineering and can represent words, sentences and documents as fix-length vectors, which contain rich semantic information and are ideal for subsequent NLP tasks.

One formidable constraint of deep neural networks (DNN) is their strong reliance on large amounts of annotated corpus due to substantial parameters to train. A DNN trained on limited data is prone to overfitting and incapable to generalize well. However, constructions of large-scale high-quality labeled datasets are extremely labor-intensive. To solve the problem, these models usually employ a pre-trained lookup table, also known as *Word Embedding* [Mikolov *et al.*, 2013b], to map words into vectors with semantic implications. However, this method just introduces extra knowledge and does not directly optimize the targeted task. The problem of insufficient annotated resources is not solved either.

Multi-task learning leverages potential correlations among related tasks to extract common features, increase corpus size implicitly and yield classification improvements. Inspired by [Caruana, 1997], there are a large literature dedicated for multi-task learning with neural network based models [Collobert and Weston, 2008; Liu *et al.*, 2015b; 2016b; 2016c]. These models basically share some lower layers to capture common features and further feed them to subsequent task-specific layers, which can be classified into three types:

- **Type-I** One dataset annotated with multiple labels and one input with multiple outputs.

- **Type-II** Multiple datasets with respective labels and one input with multiple outputs, where samples from different tasks are fed one by one into the models sequentially.

- **Type-III** Multiple datasets with respective labels and multiple inputs with multiple outputs, where samples from different tasks are jointly learned in parallel.

In this paper, we propose a generalized multi-task learning architecture with four types of recurrent neural layers for text classification. The architecture focuses on **Type-III**, which involves more complicated interactions but has not been researched yet. All the related tasks are jointly integrated into a single system and samples from different tasks are trained in parallel. In our model, every two tasks can directly interact with each other and selectively absorb useful information, or communicate indirectly via a shared intermediate layer. We also design a global memory storage to share common features and collect interactions among all tasks.

We conduct extensive experiments on five benchmark datasets for text classification. Compared to learning separately, jointly learning multiple relative tasks in our model demonstrate significant performance gains for each task.

Our contributions are three-folds:

- Our model is structurally flexible and considers various interactions, which can be concluded as a generalized case of many previous works with deliberate designs.

- Our model allows for interactions among three or more tasks simultaneously and samples from different tasks are trained in parallel with multiple inputs.

- We consider different scenarios of multi-task learning and demonstrate strong results on several benchmark classification datasets. Our model outperforms most of state-of-the-art baselines.

2 Problem Statements

2.1 Single-Task Learning

For a single supervised text classification task, the input is a word sequences denoted by $x = \{x_1, x_2, ..., x_T\}$, and the output is the corresponding class label y or class distribution $\mathbf{y}$. A lookup layer is used first to get the vector representation $\mathbf{x}_i \in \mathbb{R}^d$ of each word x_i. A classification model f is trained to transform each $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T\}$ into a predicted distribution $\hat{\mathbf{y}}$.

$$f(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T) = \hat{\mathbf{y}} \tag{1}$$

and the training objective is to minimize the total cross-entropy of the predicted and true distributions over all samples.

$$L = -\sum_{i=1}^{N} \sum_{j=1}^{C} y_{ij} \log \hat{y}_{ij} \tag{2}$$

where N denotes the number of training samples and C is the class number.

2.2 Multi-Task Learning

Given K supervised text classification tasks, $T_1, T_2, ..., T_K$, a jointly learning model $\mathbf{F}$ is trained to transform multiple inputs into a combination of predicted distributions in parallel.

$$\mathbf{F}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(K)}) = (\hat{\mathbf{y}}^{(1)}, \hat{\mathbf{y}}^{(2)}, ..., \hat{\mathbf{y}}^{(K)}) \tag{3}$$

where $\mathbf{x}^{(k)}$ are sequences from each tasks and $\hat{\mathbf{y}}^{(k)}$ are the corresponding predictions.

The overall training objective of $\mathbf{F}$ is to minimize the weighted linear combination of costs for all tasks.

$$\mathbf{L} = -\sum_{i=1}^{N} \sum_{k=1}^{K} \lambda_k \sum_{j=1}^{C_k} y_{ij}^{(k)} \log \hat{y}_{ij}^{(k)} \tag{4}$$

where N denotes the number of sample collections, C_k and λ_k are class numbers and weights for each task T_k respectively.

2.3 Three Perspectives of Multi-Task Learning

Different tasks may differ in characteristics of the word sequences $\mathbf{x}$ or the labels $\mathbf{y}$. We compare lots of benchmark tasks for text classification and conclude three different perspectives of multi-task learning.

- **Multi-Cardinality** Tasks are similar except for cardinality parameters, for example, movie review datasets with different average sequence lengths and class numbers.

- **Multi-Domain** Tasks involve contents of different domains, for example, product review datasets on books, DVDs, electronics and kitchen appliances.

- **Multi-Objective** Tasks are designed for different objectives, for example, sentiment analysis, topics classification and question type judgment.

The simplest multi-task learning scenario is that all tasks share the same cardinality, domain and objective, while come from different sources, so it is intuitive that they can obtain useful information from each other. However, in the most complex scenario, tasks may vary in cardinality, domain and even objective, where the interactions among different tasks can be quite complicated and implicit. We will evaluate our model on different scenarios in the Experiment section.

3 Methodology

Recently neural network based models have obtained substantial interests in many natural language processing tasks for their capabilities to represent variable-length text sequences as fix-length vectors, for example, Neural Bag-of-Words (NBOW), Recurrent Neural Networks (RNN), Recursive Neural Networks (RecNN) and Convolutional Neural Network (CNN). Most of them first map sequences of words, n-grams or other semantic units into embedding representations with a pre-trained lookup table, then fuse these vectors with different architectures of neural networks, and finally utilize a softmax layer to predict categorical distribution for specific classification tasks. For recurrent neural network, input vectors are absorbed one by one in a recurrent way, which makes RNN particularly suitable for natural language processing tasks.

3.1 Recurrent Neural Network

A recurrent neural network maintains a internal hidden state vector $\mathbf{h}_t$ that is recurrently updated by a transition function f. At each time step t, the hidden state $\mathbf{h}_t$ is updated according to the current input vector $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{t-1}$.

$$\mathbf{h}_t = \begin{cases} 0 & t = 0 \\ f(\mathbf{h}_{t-1}, \mathbf{x}_t) & \text{otherwise} \end{cases} \tag{5}$$

where f is usually a composition of an element-wise nonlinearity with an affine transformation of both $\mathbf{x}_t$ and $\mathbf{h}_{t-1}$.

In this way, recurrent neural networks can comprehend a sequence of arbitrary length into a fix-length vector and feed it to a softmax layer for text classification or other NLP tasks. However, gradient vector of f can grow or decay exponentially over long sequences during training, also known as the *gradient exploding* or *vanishing* problems, which makes it difficult to learn long-term dependencies and correlations for RNNs.

[Hochreiter and Schmidhuber, 1997] proposed Long Short-Term Memory Network (LSTM) to tackle the above problems. Apart from the internal hidden state $\mathbf{h}_t$, LSTM also maintains a internal hidden memory cell and three gating mechanisms. While there are numerous variants of the standard LSTM, here we follow the implementation of [Graves, 2013]. At each time step t, states of the LSTM can be fully represented by five vectors in $\mathbb{R}^n$, an *input gate* $\mathbf{i}_t$, a *forget gate* $\mathbf{f}_t$, an *output gate* $\mathbf{o}_t$, the *hidden state* $\mathbf{h}_t$ and the *memory cell* $\mathbf{c}_t$, which adhere to the following transition functions.

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{V}_i \mathbf{c}_{t-1} + \mathbf{b}_i) \tag{6}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{V}_f \mathbf{c}_{t-1} + \mathbf{b}_f) \tag{7}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{V}_o \mathbf{c}_{t-1} + \mathbf{b}_o) \tag{8}$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1}) \tag{9}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \tag{10}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{11}$$

where $\mathbf{x}_t$ is the current input, σ denotes logistic sigmoid function and $\odot$ denotes element-wise multiplication. By selectively controlling portions of the memory cell $\mathbf{c}_t$ to update, erase and forget at each time step, LSTM can better comprehend long-term dependencies with respect to labels of the whole sequences.

3.2 A Generalized Architecture

Based on the LSTM implementation of [Graves, 2013], we propose a generalized multi-task learning architecture for text classification with four types of recurrent neural layers to convey information inside and among tasks. Figure 1 illustrates the structure design and information flows of our model, where three tasks are jointly learned in parallel.

As Figure 1a shows, each task owns a LSTM-based **Single Layer** for intra-task learning. Pair-wise **Coupling Layer** and **Local Fusion Layer** are designed for direct and indirect inter-task interactions. And we further utilize a **Global Fusion Layer** to maintain a global memory for information shared among all tasks.

Single Layer

Each task owns a LSTM-based Single Layer with a collection of parameters $\Phi^{(k)}$, taking Eqs.(9) for example.

$$\tilde{\mathbf{c}}_t^{(k)} = \tanh(\mathbf{W}_c^{(k)}\mathbf{x}_t^{(k)} + \mathbf{U}_c^{(k)}\mathbf{h}_{t-1}^{(k)}) \quad (12)$$

Input sequences of each task are transformed into vector representations $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(K)})$, which are later recurrently fed into the corresponding Single Layers. The hidden states at the last time step $\mathbf{h}_T^{(k)}$ of each Single Layer can be regarded as fix-length representations of the whole sequences, which are followed by a fully connected layer and a softmax non-linear layer to produce class distributions.

$$\hat{\mathbf{y}}^{(k)} = softmax(\mathbf{W}^{(k)}\mathbf{h}_T^{(k)} + \mathbf{b}^{(k)}) \quad (13)$$

where $\hat{\mathbf{y}}^{(k)}$ is the predicted class distribution for $\mathbf{x}^{(k)}$.

Coupling Layer

Besides Single Layers, we design Coupling Layers to model **direct** pair-wise interactions between tasks. For each pair of tasks, hidden states and memory cells of the Single Layers can obtain extra information directly from each other, as shown in Figure 1b.

We re-define Eqs.(12) and utilize a gating mechanism to control the portion of information flows from one task to another. The memory content $\tilde{\mathbf{c}}_t^{(k)}$ of each Single Layer is updated on the leverage of pair-wise couplings.

$$\tilde{\mathbf{c}}_t^{(k)} = \tanh(\mathbf{W}_c^{(k)}\mathbf{x}_t^{(k)} + \sum_{j=1}^{K} \mathbf{g}^{(j \to k)}\mathbf{U}_c^{(j \to k)}\mathbf{h}_{t-1}^{(j)}) \quad (14)$$

$$\mathbf{g}^{(j \to k)} = \sigma(\mathbf{W}_{gc}^{(k)}\mathbf{x}_t^{(k)} + \mathbf{U}_{gc}^{(j)}\mathbf{h}_{t-1}^{(j)}) \quad (15)$$

where $\mathbf{g}^{(j \to k)}$ controls the portion of information flow from T_j to T_k, based on the correlation strength between $\mathbf{x}_t^{(k)}$ and $\mathbf{h}_{t-1}^{(j)}$ at the current time step.
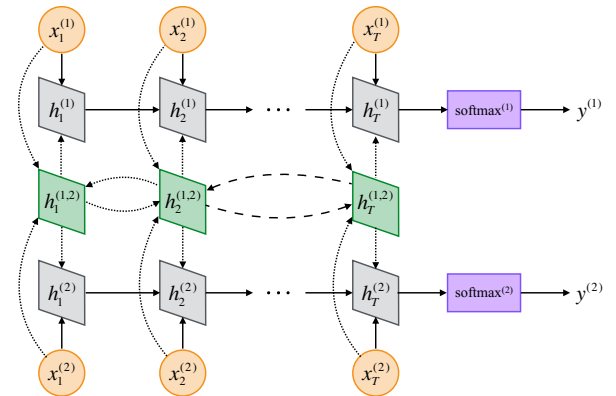
In this way, the hidden states and memory cells of each Single Layer can obtain extra information from other tasks and stronger relevance results in higher chances of reception.
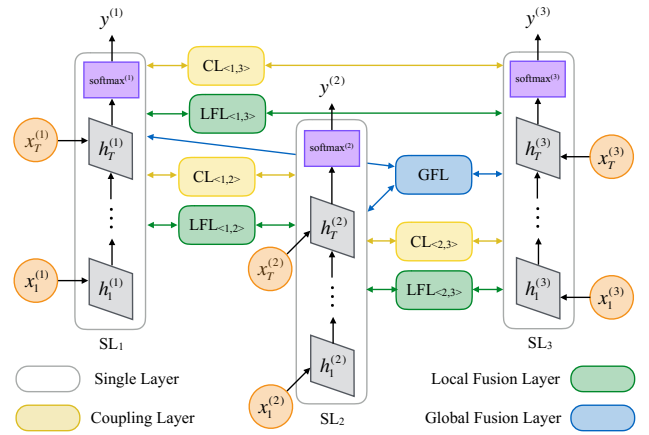
(a) Overall architecture with Single Layers, Coupling Layers, Local Fusion Layers and Global Fusion Layer
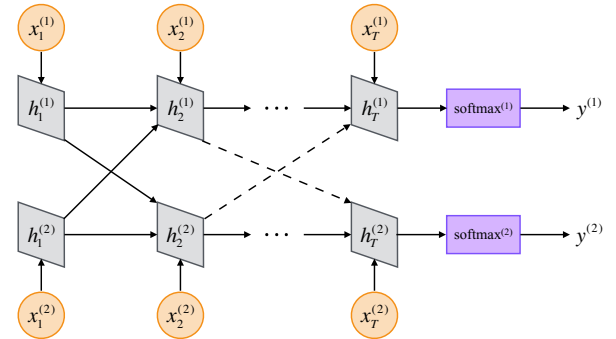
(b) Details of Coupling Layer Between T_1 and T_2

(c) Details of Local Fusion Layer Between T_1 and T_2

Figure 1: A generalized recurrent neural architecture for modeling text with multi-task learning

Local Fusion Layer

Different from Coupling Layers, Local Fusion Layers introduce a shared bi-directional LSTM Layer to model **indirect** pair-wise interactions between tasks. For each pair of tasks, we feed the Local Fusion Layer with the concatenation of both inputs, $\mathbf{x}_t^{(j,k)} = \mathbf{x}_t^{(j)} \oplus \mathbf{x}_t^{(k)}$, as shown in Figure 1c. We denote the output of the Local Fusion Layer as

$\mathbf{h}_t^{(j,k)} = \overrightarrow{\mathbf{h}}_t^{(j,k)} \oplus \overleftarrow{\mathbf{h}}_t^{(j,k)}$, a concatenation of hidden states from the forward and backward LSTM at each time step.

Similar to Coupling Layers, hidden states and memory cells of the Single Layers can selectively decide how much information to accept from the pair-wise Local Fusion Layers. We re-define Eqs.(14) by considering the interactions between the memory content $\tilde{\mathbf{c}}_t^{(k)}$ and outputs of the Local Fusion Layers as follows.

$$\tilde{\mathbf{c}}_t^{(k)} = \tanh(\mathbf{W}_c^{(k)}\mathbf{x}_t^{(k)} + \mathbf{C}_t^{(k)} + \mathbf{LF}_t^{(k)}) \qquad (16)$$

$$\mathbf{LF}_t^{(k)} = \sum_{j=1,j\neq k}^{K} \mathbf{g}^{(j,k)}\mathbf{U}_c^{(j,k)}\mathbf{h}_t^{(j,k)} \qquad (17)$$

$$\mathbf{g}^{(j,k)} = \sigma(\mathbf{W}_{gf}^{(k)}\mathbf{x}_t^{(k)} + \mathbf{U}_{gf}^{(j)}\mathbf{h}_t^{(j,k)}) \qquad (18)$$

where $\mathbf{C}_t^{(k)}$ denotes the coupling term in Eqs.(14) and $\mathbf{LF}_t^{(k)}$ represents the local fusion term. Again, we employ a gating mechanism $\mathbf{g}^{(j,k)}$ to control the portion of information flow from the Local Coupling Layers to T_k.

Global Fusion Layer
Indirect interactions between Single Layers can be **pair-wise** or **global**, so we further propose the Global Fusion Layer as a shared memory storage among all tasks. The Global Fusion Layer consists of a bi-directional LSTM Layer with the inputs $\mathbf{x}_t^{(g)} = \mathbf{x}_t^{(1)} \oplus \mathbf{x}_t^{(2)} \oplus \cdots \oplus \mathbf{x}_t^{(K)}$ and the outputs $\mathbf{h}_t^{(g)} = \overrightarrow{\mathbf{h}}_t^{(g)} \oplus \overleftarrow{\mathbf{h}}_t^{(g)}$.

We denote the global fusion term as $\mathbf{GF}_t^{(k)}$ and the memory content $\tilde{\mathbf{c}}_t^{(k)}$ is calculated as follows.

$$\tilde{\mathbf{c}}_t^{(k)} = \tanh(\mathbf{W}_c^{(k)}\mathbf{x}_t^{(k)} + \mathbf{C}_t^{(k)} + \mathbf{LF}_t^{(k)} + \mathbf{GF}_t^{(k)}) \quad (19)$$

$$\mathbf{GF}_t^{(k)} = \sigma(\mathbf{W}_{gg}^{(k)}\mathbf{x}_t^{(k)} + \mathbf{U}_{gg}^{(k)}\mathbf{h}_t^{(g)})\mathbf{U}_c^{(g)}\mathbf{h}_t^{(g)} \qquad (20)$$

As a result, our architecture covers complicated interactions among different tasks. It is capable of mapping a collection of input sequences from different tasks into a combination of predicted class distributions in parallel, as shown in Eqs.(3).

3.3 Sampling & Training

Most previous multi-task learning models [Collobert and Weston, 2008; Liu *et al.*, 2015b; 2016b; 2016c] belongs to **Type-I** or **Type-II**. The total number of input samples is $N = \sum_{k=1}^{K} N_k$, where N_k are the sample numbers of each task.

However, our model focuses on **Type-III** and requires a 4-D tensor $N \times K \times T \times d$ as inputs, where N, K, T, d are total number of input collections, task number, sequence length and embedding size respectively. Samples from different tasks are jointly learned in parallel so the total number of all possible input collections is $N_{max} = \prod_{k=1}^{K} N_k$. We propose a **Task Oriented Sampling** algorithm to generate sample collections for improvements of a specific task T_k.

Given the generated sequence collections $\mathbf{X}$ and label combinations $\mathbf{Y}$, the overall loss function can be calculated based on Eqs.(4) and (13). The training process is conducted in a

---

Algorithm 1 **T**ask **O**riented **S**ampling

Input: N_i samples from each task T_i; k, the oriented task index; n_0, upsampling coefficient s.t. $N = n_0 N_k$
Output: sequence collections $\mathbf{X}$ and label combinations $\mathbf{Y}$
1: **for** each $i \in [1, K]$ **do**
2:      generate a set S_i with N samples for each task:
3:      **if** $i = k$ **then**
4:          repeat each sample for n_0 times
5:      **else if** $N_i \geq N$ **then**
6:          randomly select N samples without replacements
7:      **else**
8:          randomly select N samples with replacements
9:      **end if**
10: **end for**
11: **for** each $j \in [1, N]$ **do**
12:      randomly select a sample from each S_i without replacements
13:      combine their features and labels as X_j and Y_j
14: **end for**
15: merge all X_j and Y_j to produce the sequence collections $\mathbf{X}$ and label combinations $\mathbf{Y}$

---

stochastic manner until convergence. For each loop, we randomly select a collection from the N candidates and update the parameters by taking a gradient step.

4 Experiment

In this section, we design three different scenarios of multi-task learning based on five benchmark datasets for text classification. we investigate the empirical performances of our model and compare it to existing state-of-the-art models.

4.1 Datasets

As Table 1 shows, we select five benchmark datasets for text classification and design three experiment scenarios to evaluate the performances of our model.

- **Multi-Cardinality** Movie review datasets with different average lengths and class numbers, including **SST-1** [Socher *et al.*, 2013], **SST-2** and **IMDB** [Maas *et al.*, 2011].

- **Multi-Domain** Product review datasets on different domains from *Multi-Domain Sentiment Dataset* [Blitzer *et al.*, 2007], including **Books**, **DVDs**, **Electronics** and **Kitchen**.

- **Multi-Objective** Classification datasets with different objectives, including **IMDB**, **RN** [Apté *et al.*, 1994] and **QC** [Li and Roth, 2002].

4.2 Hyperparameters and Training

The whole network is trained through back propagation with stochastic gradient descent [Amari, 1993]. We obtain a pre-trained lookup table by applying *Word2Vec* [Mikolov *et al.*, 2013a] on the Google News corpus, which contains more than 100B words with a vocabulary size of about 3M. All involved parameters are randomly initialized from a truncated normal distribution with zero mean and standard deviation.

Table 1: Five benchmark classification datasets: SST, IMDB, MDSD, RN, QC.

Dataset	Description	Type	Length	Class	Objective
SST	Movie reviews in Stanford Sentiment Treebank including SST-1 and SST-2	Sentence	19 / 19	5 / 2	Sentiment
IMDB	Internet Movie Database	Document	279	2	Sentiment
MDSD	Product reviews on books, DVDs, electronics and kitchen appliances	Document	176 / 189 / 115 / 97	2	Sentiment
RN	Reuters Newswire topics classification	Document	146	46	Topics
QC	Question Classification	Sentence	10	6	Question Types

For each task T_k, we conduct **TOS** with $n_0 = 2$ to improve its performance. After training our model on the generated sample collections, we evaluate the performance of task T_k by comparing $\hat{\mathbf{y}}^{(k)}$ and $\mathbf{y}^{(k)}$ on the test set. We apply 10-fold cross-validation and different combinations of hyperparameters are investigated, of which the best one, as shown in Table 2, is reserved for comparisons with state-of-the-art models.

Table 2: Hyperparameter settings

Embedding size	$d = 300$
Hidden layer size of LSTM	$n = 100$
Initial learning rate	$\eta = 0.1$
Regularization weight	$\lambda = 10^{-5}$

4.3 Results

We compare performances of our model with the implementation of [Graves, 2013] and the results are shown in Table 3. Our model obtains better performances in Multi-Domain scenario with an average improvement of 4.5%, where datasets are product reviews on different domains with similar sequence lengths and the same class number, thus producing stronger correlations. Multi-Cardinality scenario also achieves significant improvements of 2.77% on average, where datasets are movie reviews with different cardinalities.

However, Multi-Objective scenario benefits less from multi-task learning due to lacks of salient correlation among sentiment, topic and question type. The QC dataset aims to classify each question into six categories and its performance even gets worse, which may be caused by potential noises introduced by other tasks. In practice, the structure of our model is flexible, as couplings and fusions between some empirically unrelated tasks can be removed to alleviate computation costs.

Influences of n_0 in TOS
We further explore the influence of n_0 in TOS on our model, which can be any positive integer. A higher value means larger and more various samples combinations, while requires higher computation costs.

Figure 2 shows the performances of datasets in Multi-Domain scenario with different n_0. Compared to $n_0 = 1$, our model can achieve considerable improvements when $n_0 = 2$ as more samples combinations are available. However, there are no more salient gains as n_0 gets larger and potential noises from other tasks may lead to performance degradations. For a

trade-off between efficiency and effectiveness, we determine $n_0 = 2$ as the optimal value for our experiments.
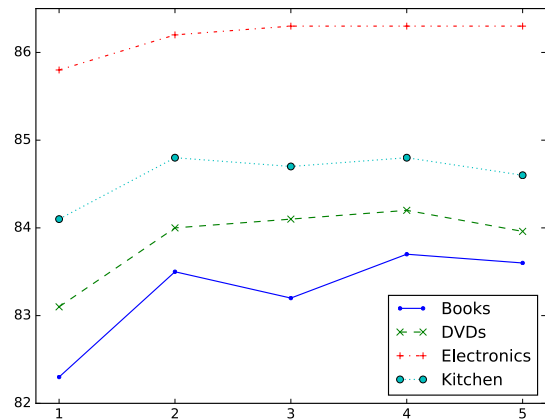

Figure 2: Influences of n_0 in TOS on different datasets

Pair-wise Performance Gain
In order to measure the correlation strength between two task T_i and T_j, we learn them jointly with our model and define Pair-wise Performance Gain as $PPG_{ij} = \sqrt{\frac{P_i' P_j'}{P_i P_j}}$, where P_i, P_j, P_i', P_j' are the performances of tasks T_i and T_j when learned individually and jointly.

We calculate PPGs for every two tasks in Table 1 and illustrate the results in Figure 3, where darkness of colors indicate strength of correlation. It is intuitive that datasets of Multi-Domain scenario obtain relatively higher PPGs with each other as they share similar cardinalities and abundant low-level linguistic characteristics. Sentences of QC dataset are much shorter and convey unique characteristics from other tasks, thus resulting in quite lower PPGs.

4.4 Comparisons with State-of-the-art Models

We apply the optimal hyperparameter settings and compare our model against the following state-of-the-art models:

- **NBOW** Neural Bag-of-Words that simply sums up embedding vectors of all words.

- **PV** Paragraph Vectors followed by logistic regression [Le and Mikolov, 2014].

- **MT-RNN** Multi-Task learning with Recurrent Neural Networks by a shared-layer architecture [Liu *et al.*, 2016c].

Table 3: Results of our model on different scenarios

Model	Multi-Cardinality			Multi-Domain				Multi-Objective		
	SST-1	SST-2	IMDB	Books	DVDs	Electronics	Kitchen	IMDB	RN	QC
Single Task	45.9	85.8	88.5	78.0	79.5	81.2	81.8	88.5	83.6	**92.5**
Our Model	**49.2**	**87.7**	**91.6**	**83.5**	**84.0**	**86.2**	**84.8**	**89.7**	**84.2**	92.3

Table 4: Comparisons with state-of-the-art models

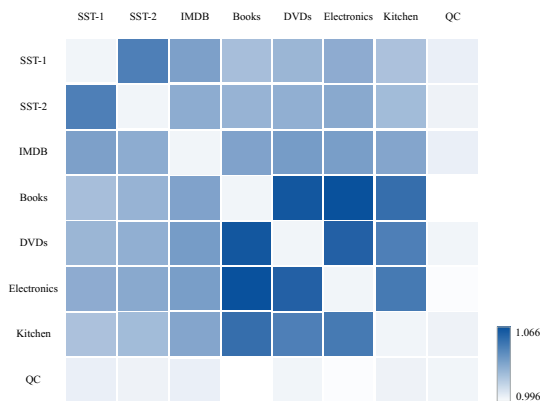| Model	SST-1	SST-2	IMDB	Books	DVDs	Electronics	Kitchen	QC
NBOW	42.4	80.5	83.62	-	-	-	-	88.2
PV	44.6	82.7	**91.7**	-	-	-	-	91.8
MT-RNN	**49.6**	**87.9**	91.3	-	-	-	-	-
MT-CNN	-	-	-	80.2	81.0	83.4	83.0	-
MT-DNN	-	-	-	79.7	80.5	82.5	82.8	-
GRNN	47.5	85.5	-	-	-	-	-	**93.8**
Our Model	49.2	87.7	91.6	**83.5**	**84.0**	**86.2**	**84.8**	92.3

Figure 3: Visualization of Pair-wise Performance Gains

- **MT-CNN** Multi-Task learning with Convolutional Neural Networks [Collobert and Weston, 2008] where lookup tables are partially shared.
- **MT-DNN** Multi-Task learning with Deep Neural Networks [Liu *et al.*, 2015b] that utilizes bag-of-word representations and a hidden shared layer.
- **GRNN** Gated Recursive Neural Network for sentence modeling [Chen *et al.*, 2015].

As Table 4 shows, our model obtains competitive or better performances on all tasks except for the QC dataset, as it contains poor correlations with other tasks. MT-RNN slightly outperforms our model on SST, as sentences from this dataset are much shorter than those from IMDB and MDSD, and another possible reason may be that our model are more complex and requires larger data for training. Our model proposes the designs of various interactions including coupling, local and global fusion, which can be further implemented by other state-of-the-art models and produce better performances.

5 Related Work

There are a large body of literatures related to multi-task learning with neural networks in NLP [Collobert and Weston,

2008; Liu *et al.*, 2015b; 2016b; 2016c].

[Collobert and Weston, 2008] belongs to **Type-I** and utilizes shared lookup tables for common features, followed by task-specific neural layers for several traditional NLP tasks such as part-of-speech tagging and semantic parsing. They use a fix-size window to solve the problem of variable-length texts, which can be better handled by recurrent neural networks.

[Liu *et al.*, 2015b; 2016b; 2016c] all belong to **Type-II** where samples from different tasks are learned sequentially. [Liu *et al.*, 2015b] applies bag-of-word representation and information of word orders are lost. [Liu *et al.*, 2016b] introduces an external memory for information sharing with a reading/writing mechanism for communicating, and [Liu *et al.*, 2016c] proposes three different models for multi-task learning with recurrent neural networks. However, models of these two papers only involve pair-wise interactions, which can be regarded as specific implementations of Coupling Layer and Fusion Layer in our model.

Different from the above models, our model focuses on **Type-III** and utilize recurrent neural networks to comprehensively capture various interactions among tasks, both direct and indirect, local and global. Three or more tasks are learned simultaneously and samples from different tasks are trained in parallel benefitting from each other, thus obtaining better sentence representations.

6 Conclusion and Future Work

In this paper, we propose a multi-task learning architecture for text classification with four types of recurrent neural layers. The architecture is structurally flexible and can be regarded as a generalized case of many previous works with deliberate designs. We explore three different scenarios of multi-task learning and our model can improve performances of most tasks with additional related information from others in all scenarios.

In future work, we would like to investigate further implementations of couplings and fusions, and conclude more multi-task learning perspectives.

References

[Amari, 1993] Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(3):185–196, 1993.

[Apté *et al.*, 1994] Chidanand Apté, Fred Damerau, and Sholom M. Weiss. Automated Learning of Decision Rules for Text Categorization. *ACM Trans. Inf. Syst.*, 12(3):233–251, 1994.

[Bengio *et al.*, 2013] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, 2013.

[Blitzer *et al.*, 2007] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *ACL*, 2007.

[Caruana, 1997] Rich Caruana. Multitask Learning. *Machine Learning*, 28(1):41–75, 1997.

[Chen *et al.*, 2015] Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Shiyu Wu, and Xuanjing Huang. Sentence Modeling with Gated Recursive Neural Network. In *EMNLP*, pages 793–798, 2015.

[Collobert and Weston, 2008] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *ICML*, pages 160–167, 2008.

[Graves, 2013] Alex Graves. Generating Sequences With Recurrent Neural Networks. *CoRR*, abs/1308.0850, 2013.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[Le and Mikolov, 2014] Quoc V. Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1188–1196, 2014.

[Li and Roth, 2002] Xin Li and Dan Roth. Learning Question Classifiers. In *COLING*, 2002.

[Liu *et al.*, 2015a] Pengfei Liu, Xipeng Qiu, Xinchi Chen, Shiyu Wu, and Xuanjing Huang. Multi-Timescale Long Short-Term Memory Neural Network for Modelling Sentences and Documents. In *EMNLP*, pages 2326–2335, 2015.

[Liu *et al.*, 2015b] Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. Representation Learning Using Multi-Task Deep Neural Networks for Semantic Classification and Information Retrieval. In *NAACL HLT*, pages 912–921, 2015.

[Liu *et al.*, 2016a] Pengfei Liu, Xipeng Qiu, Jifan Chen, and Xuanjing Huang. Deep Fusion LSTMs for Text Semantic Matching. In *ACL*, 2016.

[Liu *et al.*, 2016b] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Deep Multi-Task Learning with Shared Memory for Text Classification. In *EMNLP*, pages 118–127, 2016.

[Liu *et al.*, 2016c] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent Neural Network for Text Classification with Multi-Task Learning. In *IJCAI*, pages 2873–2879, 2016.

[Liu *et al.*, 2016d] Pengfei Liu, Xipeng Qiu, Yaqian Zhou, Jifan Chen, and Xuanjing Huang. Modelling Interaction of Sentence Pair with Coupled-LSTMs. In *EMNLP*, pages 1703–1712, 2016.

[Maas *et al.*, 2011] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In *NAACL HLT*, pages 142–150. Association for Computational Linguistics, June 2011.

[Mikolov *et al.*, 2013a] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781, 2013.

[Mikolov *et al.*, 2013b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[Socher *et al.*, 2013] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *EMNLP*, pages 1631–1642, Stroudsburg, PA, October 2013. Association for Computational Linguistics.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.