Sentiment Classification by Leveraging the Shared Knowledge from a Sequence of Domains

Guangyi Lv[1], Shuai Wang[2], Bing Liu[2], Enhong Chen[1(✉)], and Kun Zhang[1]

[1] Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China, Hefei, China
{gylv,zhkun}@mail.ustc.edu.cn, cheneh@ustc.edu.cn
[2] University of Illinois at Chicago, Chicago, USA
shuaiwanghk@gmail.com, liub@uic.edu

Abstract. This paper studies sentiment classification in a setting where a sequence of classification tasks is performed over time. The goal is to leverage the knowledge gained from previous tasks to do better on the new task than without using the previous knowledge. This is a lifelong learning setting. This paper proposes a novel deep learning model for lifelong sentiment classification. The key novelty of the proposed model is that it uses two networks: a *knowledge retention network* for retaining domain-specific knowledge learned in the past, and a feature learning network for classification feature learning. The two networks work together to perform the classification task. Our experimental results show that the proposed deep learning model outperforms the state-of-the-art baselines.

1 Introduction

In many sentiment analysis applications, one needs to perform many tasks over time, e.g., to analyze reviews of different products for many different clients, or to analyze reviews of many/all products of a single client (e.g., a retailer). A natural question is whether the system can improve itself over time as it analyzes reviews of more and more categories of products. This is a *lifelong learning* (LL) setting. LL is stated as follows: At any point in time, the learner has performed learning on a sequence of tasks from 1 to $N-1$. When faced with the Nth task, it uses the knowledge gained in the past $N-1$ tasks to help learn the Nth task [7,31,34], which should do better than learning using only the training data of the Nth task alone (without using any past knowledge). In our case, each task is a sentiment classification problem that aims to classify each review into the positive or negative class [19,23].

In [8], a lifelong learning technique is proposed for sentiment classification in the context of naive Bayesian Classification. The method exploited the knowledge of word probabilities under different classes in the past tasks/domains as priors to help optimize the new task learning. In this paper, we propose a novel

G. Li et al. (Eds.): DASFAA 2019, LNCS 11446, pp. 795–811, 2019.
https://doi.org/10.1007/978-3-030-18576-3_47

deep learning model, which outperforms this existing approach. Note that by a domain we mean a category of products. Since each of our task is from a different domain, we use the terms *domain* and *task* interchangeably in this paper.

The proposed deep learning architecture is called SRK (***S****entiment classification by leveraging the* ***R****etained* ***K****nowledge*). It consists of two networks, i.e., the "Feature Learning Network" (FLN) and the "Knowledge Retention Network" (KRN), which jointly perform the supervised learning task. The FLN is like a traditional network for supervised learning. The only difference is that it learns a sequence of tasks. When it comes to learn the next/new task, it has a set of very rich features as the network parameters gained from the previous tasks can help the new task learning. That is because sentiment classification in different domains is quite similar in the sense that sentiment expressions are largely shared and have the same polarities across domains.

As we will see in Sect. 4.2, FLN alone is already able to outperform learning using only the data from each task (without using any previous knowledge or task data). However, we can do better. One weakness with using FLN alone is that if the immediate previous task $N-1$ is very different from the new task N, then the new task may not perform well (see Sect. 4.3) because the short-term memorized/residual parameters from task $N-1$, which can be viewed as a type of parameter initialization, will not be helpful to the new task N. But there may be some other earlier tasks that may be helpful to the new task N. However, it is hard for FLN to use them because of *catastrophic forgetting* [10]. Catastrophic forgetting describes a phenomenon in learning multiple tasks sequentially in a neural network, i.e., after each new task is learned, the knowledge learned from the previous tasks may be forgotten by the network. Because of this problem, the past knowledge that can be leveraged by the new task in FLN is quite limited, which results in weaker performance (see Sect. 4.3). To deal with this problem, we introduce the knowledge retention network KRN with the goal of retaining domain-specific knowledge from the past domains. In order to achieve this goal, KRN needs to deal with catastrophic forgetting.

A novel learning mechanism is proposed for KRN to deal with the forgetting problem. It is thus able to retain domain-specific knowledge learned from individual past domains, which is very helpful in learning similar tasks in the future. The two networks (FLN and KRN) work together through a gate mechanism to provide a robust solution.

Although several models for dealing with catastrophic forgetting exist, their main goal is to preserve the past task learning. That is, after learning the Nth task, the previous $N-1$ tasks will still work. Their main goal is not to leverage the knowledge learned and retained from the previous tasks to help learn the new Nth task better. The focus of the proposed architecture is to leverage the past learned knowledge to do better for the new task. That is why we propose a two-network solution. More related work will be discussed in Sect. 2.

In summary, this paper makes the following contributions:

1. It proposes a deep learning architecture SRK for lifelong sentiment classification, which has not been done before. SRK consists of two sub-networks to

separately capture rich features and also shared knowledge of tasks. A knowledge gate is designed to make the two networks work jointly. As discussed above, this architecture allows the system to flexibly leverages the knowledge learned from previous domains in the new domain.
2. To achieve the goal of retaining domain-specific knowledge to be used in similar tasks in the future, the knowledge retention network (KRN) needs to deal with catastrophic forgetting. A novel partially (gradient) update mechanism is proposed for the purpose based on the observation of activation sparsity in neural networks (see Sect. 3).
3. Experimental results on sentiment classification show that the proposed model SRK outperforms the state-of-the-art baselines from both lifelong learning and continual learning which tries to deal with catastrophic forgetting. It is also more stable/robust as we will see in Sect. 4.3.

2 Related Work

Our work is related to text classification. Traditional text classification mainly uses Bag-of-Words models. In recent years, neural networks have been very popular due to their superior performance [47]. Classic neural networks do not consider the sequential information in text, which is a drawback. To address the issue, various Recurrent Neural Networks (RNNs) have been proposed, e.g., Long Short-Term Memory (LSTM) [13], Gated Recurrent Units (GRU) [9], bidirectional RNN [44], and attention-based RNN [42]. In this work, we adopted GRU as the base for our lifelong RNN model. GRU has been shown to achieve similar performances as LSTM for many NLP tasks.

2.1 Sentiment Classification

Sentiment classification, a key task of sentiment analysis [23], is a special case of text classification. Earlier techniques use hand-crafted features and external resources [21,23,38,43,46], which involve a great deal of human efforts. Recent data-driven neural network methods directly use word features and/or discovered new features by the models themselves. Many neural network based sentiment classification techniques have been reported [4,27,32,41,48]. They don't involve lifelong learning. Many aspect-based sentiment analysis methods are also reported [5,14,17,26,33,36,37,39]. However, these methods also do not accumulate or use knowledge to learn new tasks as we do.

2.2 Lifelong Learning and Catastrophic Forgetting

Our work is most closely related to lifelong learning (LL) [6,7,31,34]. LL has been used for sentiment classification in [8] based on naive Bayesian classifier. However, its method is not applicable to neural networks because LL in [8] exploits word probabilities from past domains, which is not directly usable in

neural networks. Furthermore, neural networks suffer from catastrophic forgetting [10] when it learns continually. Similarly, the method in [29] is also hard to apply to neural networks.

Several methods, under the name of *continual learning*, have been proposed to overcome catastrophic forgetting, e.g., system-level consolidation [15,16] and sequential Bayesian [2,11]. However, their main goal is to preserve the previously learned results for past tasks while learning new tasks. Our focus is on how to exploit previously learned knowledge to improve the new task learning. Also, existing works are independent models with specific loss functions. They are hard to be employed in our two-brunch framework.

2.3 Transfer Learning and Multitask Learning

Transfer learning or domain adaptation uses labeled training examples from the source domain to help learning in the target domain that has no or very few labeled training examples [1,12,18,22,40]. One typical use of transfer learning is deep domain adaptation, which usually setups shared weights [20,35] or two-stream architecture [28] to reduce the difference of distributions between the source and target domains. Our feature learning network (FLN) functions like transfer learning as it can transfer some knowledge from past domains to the new domain. However, its transfer is limited as discussed in Sect. 1 because the new domain is mainly affected by its immediate previous domain due to catastrophic forgetting. That is why we use the knowledge-retention network (KRN) to retain past domain-specific knowledge so that the new domain learning can use knowledge from any similar past domains, which transfer learning or domain adaptation cannot do.

LL is also different from multitask learning [3], which tries to optimize the learning of multiple tasks simultaneously [17,30,45]. It does not accumulate past knowledge and does not learn sequentially as LL does.

3 The Proposed Solution

Our lifelong sentiment classification problem can be stated as follows: At a particular point in time, the system has performed a sequence of $N-1$ sentiment classification (SC) tasks using their training data D_1 to D_{N-1}. It now has to learn to perform the Nth new SC task given its training data D_N, where each training example is labeled with a positive or negative polarity (or sentiment). The goal is to leverage the knowledge gained in the past $N-1$ tasks to produce a better classifier than without using the past knowledge. The following subsections present the proposed model.

3.1 Overall Framework

The overall architecture of the proposed SRK model is shown in Fig. 1. It consists of three main components: (1) Feature Learning Network (2) Knowledge Retention Network, and (3) Network Fusion component.

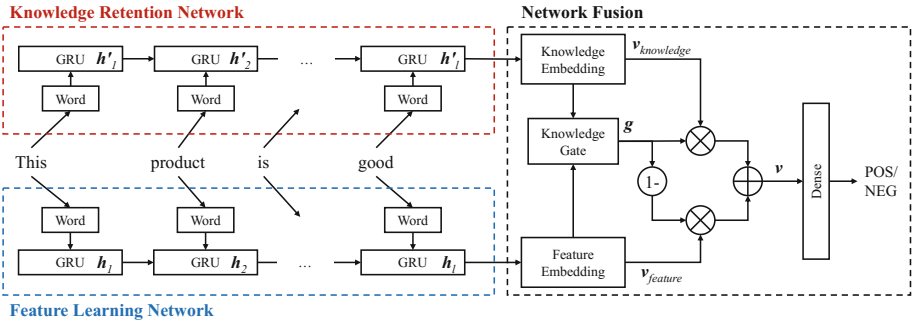

Fig. 1. The overall architecture of the proposed SRK model.

Feature Learning Network. We use the Feature Learning Network (FLN) to learn document representations to perform classification. As shown in the blue box in Fig. 1, the structure of this network is a conventional Gated Recurrent Units (GRU) [9], which is formulated as:

$$\begin{aligned} \boldsymbol{z} &= \sigma(\boldsymbol{U}_z \boldsymbol{x}_t + \boldsymbol{W}_z \boldsymbol{h}_{t-1}), \\ \boldsymbol{r} &= \sigma(\boldsymbol{U}_r \boldsymbol{x}_t + \boldsymbol{W}_r \boldsymbol{h}_{t-1}), \\ \boldsymbol{h'} &= \tanh(\boldsymbol{U}_h \boldsymbol{x}_t + \boldsymbol{W}_h (\boldsymbol{r} \odot \boldsymbol{h}_{t-1})), \\ \boldsymbol{h}_t &= (1 - \boldsymbol{z}) \odot \boldsymbol{h'} + \boldsymbol{z} \odot \boldsymbol{h}_{t-1}, \end{aligned} \tag{1}$$

whose inputs are one-hot representations of words in sentences $\boldsymbol{x} = \{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_l\}$, where l is the sentence length. These one-hot vectors are first mapped into dense representations by a pre-trained word embedding model $\boldsymbol{E}$. Then, we utilize a GRU cell whose state size is 500 to process the words one by one:

$$\begin{aligned} \boldsymbol{u}_i &= \boldsymbol{E}\boldsymbol{w}_i, \quad i \in 1, 2, \ldots, l, \\ \{\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_l\} &= \mathrm{GRU}(\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_l\}), \end{aligned} \tag{2}$$

$\{\boldsymbol{u}_i\}$ and $\{\boldsymbol{h}_i\}$ separately indicate word embedding of word i and GRU's ith state, respectively. We take the final hidden state $\boldsymbol{h}_l$ as the *feature embedding* $\boldsymbol{v}_{feature} = \boldsymbol{h}_l$ or representation of the whole document.

Knowledge Retention Network. To achieve lifelong sentiment classification, we propose a Knowledge Retention Network to learn and to retain domain-specific knowledge from previous tasks. Similar to Feature Learning Network, the structure of this network is also a GRU cell with its state size 500 which has the same input $\boldsymbol{x}$ and gives out states $\{\boldsymbol{h}'_1, \boldsymbol{h}'_2, \ldots, \boldsymbol{h}'_l\}$. The final state $\boldsymbol{h}'_l$ is used as the *knowledge embedding* $\boldsymbol{v}_{knowledge} = \boldsymbol{h}'_l$. To retain knowledge, a different learning method, called *partial update*, is proposed, which will be described in detail in Sect. 3.2.

Network Fusion Component. To make the above networks work together to produce the final result, we designed a *Knowledge Gate* to integrate the two

kinds of representations, which is shown in the black box in Fig. 1. By using the knowledge gate, the relevant information for the current task from both two networks will be selectively combined to make the final decision, which is formulated as follows:

$$\begin{aligned} \boldsymbol{g} &= \sigma(\boldsymbol{W}_k \boldsymbol{v}_{feature} + \boldsymbol{U}_k \boldsymbol{v}_{knowledge}), \\ \boldsymbol{v} &= (\boldsymbol{1} - \boldsymbol{g}) \cdot \boldsymbol{v}_{feature} + \boldsymbol{g} \cdot \boldsymbol{v}_{knowledge}, \\ P(y|\boldsymbol{x}) &= \sigma(\boldsymbol{W}\boldsymbol{v} + \boldsymbol{b}). \end{aligned} \tag{3}$$

Here, $\boldsymbol{g}$ denotes the knowledge gate, $\boldsymbol{W}$ and $\boldsymbol{b}$ are parameters of the last fully connected layer, and σ means the sigmoid activation function.

Finally, we choose Mean-Square Error (MSE) as the training loss function. Note that other loss functions such as cross-entropy can also be applied as alternatives. Furthermore, to ensure both the Feature Learning and the Knowledge Retention Networks learn good representations, we also use additional losses for those two networks:

$$\begin{aligned} L_F &= \text{MSE}(y, \sigma(\boldsymbol{W}\boldsymbol{v}_{feature} + \boldsymbol{b})), \\ L_K &= \text{MSE}(y, \sigma(\boldsymbol{W}\boldsymbol{v}_{knowledge} + \boldsymbol{b})), \end{aligned} \tag{4}$$

where y is the true label. The final objective function is formulated as:

$$L = \text{MSE}(y, P(y|\boldsymbol{x})) + L_F + L_K. \tag{5}$$

3.2 Partial Updating Mechanism

In order to perform sentiment classification tasks sequentially, the *Knowledge Retention Network* needs to learn and retain domain-specific knowledge from every task. However, conventional neural networks have poor performances because they often suffer from the catastrophic forgetting problem as we discussed in the introduction section. Based on our pilot study, we found that even though neural network based models have the ability to learn knowledge from tasks in our case, they have the tendency to remember only knowledge

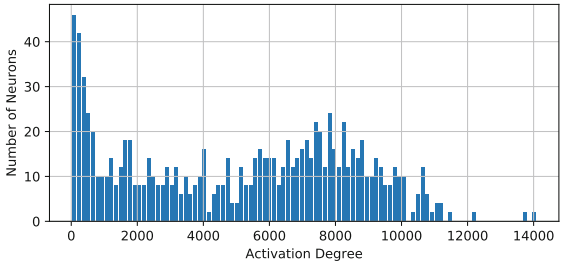

Fig. 2. Activation statistics in the state layer of GRU after convergence. Higher activation degree means more chance to be activated in a task.

that is common across all tasks rather than important specific features in specific domains. They also tend to remember more of the last task learned due to catastrophic forgetting. We propose a *Partial Update Mechanism* to solve the problem.

The idea of partial update is inspired by the observation of activation sparsity in neural networks. Figure 2 shows the histogram of the activation value of state vector $\boldsymbol{h}_t$ in Eq. 1 from a trained GRU network. We can see that only a small number of hidden nodes have very high degrees of activation. Most hidden nodes have relatively tiny activation values. This phenomenon becomes more obvious as the number of model parameters grows.

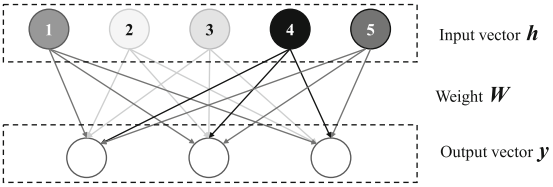

Fig. 3. An example of a dense layer from a network. Colors of the input neurons indicate the activation values. Darker color means a higher activation value or more activated.

This observation indicates that we can exploit the property of activation sparsity for knowledge learning and sharing among tasks. Specifically, for any network converged on task $(\boldsymbol{D}_i, \boldsymbol{T}_i)$, we compute such statistical information for every layer. Shown in Fig. 3, it is clear that the output weights corresponding to those less activated neurons (neurons 2 and 3) can be regarded as less important to task $(\boldsymbol{D}_i, \boldsymbol{T}_i)$. At the same time, we can say those weights corresponding to those frequently activated neurons (neurons 1, 4 and 5) store much of the important knowledge of the task. It is easy to understand that in a matrix multiplication form $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{h}$, when h_2 and h_3 have tiny values, any modification for columns 2 and 3 in matrix $\boldsymbol{W}$ have relatively little impact on the output $\boldsymbol{y}$. Thus, when we train matrix multiplication based networks like RNNs, we can keep some columns in the weight matrix unchanged, and only update those less important ones.

To simplify our description, we name the partially updated weights as *lifelong weights*, their inputs as *control vectors* and the tiny valued components of the control vectors as *free neurons*. Along this line, we can divide the partial update mechanism into two parts: **Free Neuron Detection** and **Gradient Mask**.

Free Neuron Detection. Suppose we are training a model with lifelong weights $\{\boldsymbol{W}^i\}$ and control vectors $\{\boldsymbol{h}^i\}$. To measure the importance of neurons in $\{\boldsymbol{h}^i\}$, we maintain a series of statistical information for $\{\boldsymbol{h}^i\}$ as $\{\boldsymbol{s}^i\}$, where the values of $\boldsymbol{s}^i$ indicate the activation degrees of the corresponding neurons.

For each feed forward pass, values in $\{\boldsymbol{s}^i\}$ are updated as follow:

$$\boldsymbol{s}_t^i = \boldsymbol{s}_{t-1}^i + |\boldsymbol{h}_t^i|, \quad t = 1, 2, \ldots, \quad \boldsymbol{s}_0^i = \boldsymbol{0}, \tag{6}$$

where $|\boldsymbol{h}_t^i|$ indicates the absolute value of the control vector, and $\boldsymbol{s}_t^i$ is actually the accumulation of all $|\boldsymbol{h}_t^i|$ through feed forward propagation during a task training.

As the number of tasks grows, accumulation values $\{\boldsymbol{s}^i\}$ become larger and larger. To avoid overflow, we perform a linear normalization for $\{\boldsymbol{s}^i\}$ before using it directly:

$$\hat{s}_j^i = \frac{s_j^i - \min(\boldsymbol{s}^i)}{\max(\boldsymbol{s}^i) - \min(\boldsymbol{s}^i)}, \quad j = 1, 2, \ldots, d, \tag{7}$$

where d is the dimension of $\boldsymbol{s}^i$, and min() and max() indicate the minimum and maximum values across $\boldsymbol{s}^i$ respectively. Then, in order to find free neurons based on $\hat{\boldsymbol{s}}^i$, we further sort the values of $\hat{\boldsymbol{s}}^i$ in ascending order. The dimensions (neurons) whose values are in the top ϵ percentage will be regarded as free neurons.

Gradient Mask. With the free neurons detected, when we are going to train the model on a new task, *gradient masks* will be used. A gradient mask is a vector which has the same size as the corresponding control vector $\boldsymbol{h}^i$. It tells us which columns in the lifelong weight $\boldsymbol{W}^i$ can be updated. Based on the normalized accumulation $\hat{\boldsymbol{s}}^i$ defined in Eq. 7, we compute the gradient mask $\boldsymbol{m}^i$ for weight $\boldsymbol{W}^i$ as:

$$m_j^i = \begin{cases} 1 - \hat{s}_j^i & \hat{s}_j^i \textit{ is in the top } \epsilon\%, \\ 0 & \textit{otherwise.} \end{cases} \tag{8}$$

Then the update rule for weight $\boldsymbol{W}^i$ is modified as:

$$\boldsymbol{W}^i = \boldsymbol{W}^i - \eta(\boldsymbol{m}^i \odot \frac{\partial L}{\partial \boldsymbol{W}^i}). \tag{9}$$

where η is the learning rate for back-propagation, and $\odot$ represents element-wise multiplication across the rows of the gradient matrix.

We want the ϵ to be self-adaptive in the lifelong learning setting. We propose a self-adjusting strategy for ϵ. It is intuitive that if the model starts the first task with a small ϵ, the knowledge network will suffer from the *cold start* problem, and will have poor performance for a long time. To avoid this, we set ϵ to 100% to allow the parameters to be fully updated in the first task, and then gradually decrease it to a minimum value using the following function:

$$\epsilon = (1 - \tau)e^{-\lambda n} + \tau, \quad n = 0, 1, \ldots, N, \tag{10}$$

where n is the index of the current task, λ and τ separately indicate the scale factor and the minimum value of the threshold, respectively. In our experiments, we found setting both λ and τ to 0.1 usually generate good results.

Specifically, for our *Knowledge Retention Network*, we can find in Eq. 1 that the lifelong weights are $\boldsymbol{U}^z$, $\boldsymbol{W}^z$, $\boldsymbol{U}^r$, $\boldsymbol{W}^r$, $\boldsymbol{U}^h$ and $\boldsymbol{W}^h$. The corresponding control vectors are $\boldsymbol{x}_t$, $\boldsymbol{h}_{t-1}$ and $\boldsymbol{r} \odot \boldsymbol{h}_{t-1}$. In order to increase the value difference between the neurons with high activation degrees and the neurons with low

activation degrees, i.e., to enlarge the gap of their activation values in $\boldsymbol{h}_{t-1}$ (in a GRU), we replace the original activation function tanh with the following:

$$f(x) = \begin{cases} 0.9 + kx & x \geq 0.9, \\ x & 0 < x < 0.9, \\ kx & x \leq 0, \end{cases} \tag{11}$$

which is modified from the leaky-ReLU and k is the leak value that is usually set as 0.001. We limit the value of the positive axis to 0.9 in order to make sure that in most cases, the activation value will be no larger than 1.0, which is important to avoid gradient explosion.

4 Experiments

We now evaluate the proposed model SRK and compare it with state-of-the-art baselines. We will see that SRK is not only markedly better than the baselines in general, but also more stable/robust when there is a major domain difference in the successive tasks, a highly imbalanced training data distribution, or a very small training dataset. This is because the accumulated past knowledge in SRK can compensate for these undesirable but realistic training scenarios.

4.1 Experimental Setup

Experiment Data: We use the real-world Amazon review dataset from [8]. This dataset has reviews of 20 categories of diverse products, which we also called 20 domains. Following the existing studies in [1,8,24], for each domain, the reviews with rating score greater than 3 (>3) are regarded as positive reviews and the reviews with rating score less than 3 (<3) are regarded as negative reviews. This gives us 20 sentiment classification tasks (one for each domain). Details about the dataset are given in Table 1. The size in the table is the number of reviews and the neg(%) is the proportion of negative reviews in each domain. For each domain, we further split its dataset into training and testing sets by 80% and 20%. During training, 10% of the training data is used as the development set. Note that our dataset setting here is slightly different from that in [8], which uses only 1000 reviews for each domain before the training and testing split because their method is based on naive Bayes classification. We use the full dataset of each domain for training because deep learning typically needs more training data to learn effective models.

Models for Comparison. We consider the following models for comparison.

– **SRK**. This is our proposed model. Weights of the GRUs are initialized randomly using truncated normal distribution $N(0, 0.001)$. Weights of other dense layers are randomly set following the uniform distribution in the range between $-\sqrt{6/(nin + nout)}$ and $\sqrt{6/(nin + nout)}$, where nin and $nout$ are the number of input and output neurons, respectively. Additionally, we use

Table 1. Data statistics about 20 domains

Domain	Size	Neg (%)	Domain	Size	Neg (%)
Kitchen	5,646	15.18	PC	7,393	21.58
Software	3,633	30.31	Players	5,467	43.23
Sports	4,638	15.17	Camera	4,587	29.28
Music	1,948	2.42	Tools	6,463	29.81
Baby	3,723	13.02	Audio	3,661	19.91
Home	9,355	18.9	Phone	2,703	32.89
Books	2,725	20.9	Laptop	3,686	23.28
Shoes	6,523	12.52	TV	4,579	28.41
Automotive	5,677	10.79	Network	2,815	27.32
Bed	2,750	17.32	Office	4,575	30.01

GloVe [25] as the pre-trained word embedding mentioned in Eq. 2. The model is trained using the RMSProp optimizer[1], where the batch size, learning rate and momentum are set as 32, 0.0001 and 0.9, respectively.

- **LSC** (Lifelong Sentiment Classification). This is the naïve Bayes-based lifelong sentiment classification model in [8], which accumulates knowledge in a Bayesian framework. We use all the parameter settings used in the original work. We obtained the LSC system (and also the Amazon review dataset) from the first author of the paper.
- **I-RNN** (Isolated RNN). A classic RNN model performing each task individually. It does not use knowledge sharing between multiple tasks. Each task is viewed as an isolated one. Its parameters follow the same setting as SRK.
- **FLN**. This model uses only the feature learning network (FLN) of the proposed SRK as described in Sect. 3.1. That is, only the FLN branch of SRK is employed. Its parameters also follow the same setting as SRK.
- **2-FLN**. This model uses two identical feature learning networks (FLNs). Its parameters also follow the same setting as SRK. We use this baseline to see if our SRK results can be achieved by a combination of two FLNs.
- **EWC** (Elastic Weight Consolidation). This is a well-known state-of-the-art algorithm for continual learning that deals with catastrophic forgetting [15]. EWC tries to keep the network parameters close to those of past tasks during the processing of a new task. To achieve that, a constraint is imposed on the network to make the learned parameters staying close to their old values weighted by their importance to the previous tasks. We follow the parameter settings in the original paper.

Evaluation Measure. To have a fair and clear comparison with LSC [8], which is most closely related to our work, we use balanced test sets (by down-sampling the majority class) in our testing (training still uses the original imbalanced

[1] http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.

data as in [8]). Due to the balanced test set, we use accuracy as the evaluation measure. Note that for all our experiments, all compared systems use the same training and test data.

Two Experimental Settings: We use two experimental settings to evaluate SRK: (1) Knowledge accumulation and application. In this setting, we show the importance of knowledge accumulation to the new domain task. Specifically, each domain is treated as the last domain with the rest 19 domains (randomly sequenced) as the previous (or past) domains. (2) Continuous learning and testing. In this case, we evaluate model performance in a continuous learning setting. Specifically, we first set a fixed sequence of all domains, which is randomly chosen. We then let a model process every domain one after another based on the sequence. After each model is built for a domain, it is evaluated based on the test data of that domain. We do this for all candidate models so as to test their performance in this continuous learning setting.

4.2 Results in Experimental Setting (1)

In the first experimental setting, each domain is treated as the last domain in turn. We first compare the performance from all candidate models. The accuracy scores shown in Table 2 are the averaged scores of all 20 domains (averaging the scores of each domain being the last). Based on the overall results, we can see that SRK gives the best accuracy overall.

Table 2. Overall performance of all candidate models.

Model	Accuracy	Model	Accuracy
LSC	86.68	2-FLN	88.44
I-RNN	83.86	EWC	86.29
FLN	87.73	SRK	**89.85**

Comparing SRK with I-RNN, FLN and 2-FLN, we first see that FLN improves I-RNN markedly. This is because I-RNN treats every domain as an independent task, but FLN implicitly retains some common knowledge in its parameters during learning of previous tasks. This also indicates the existence of knowledge sharing in sentiment classification tasks. Comparing 2-FLN and FLN, we see that 2-FLN improves the results obtained by FLN. This is mainly attributed to the capacity of more knowledge learned by the two networks in 2-FLN. Although 2-FLN only learns a single type of features, its two-thread learning mechanism helps capture and keep more useful information from the data. However, SRK outperforms 2-FLN. This is also intuitive because SRK models two types of different information separately, namely, domain-specific (KRN) and cross-domain knowledge (FLN), whereas 2-FLN only considers the same type of information in its two network components. Note that although

Table 3. Performance on 20 domains from all candidate models.

Domain	LSC	I-RNN	FLN	2-FLN	EWC	SRK
Kitchen	74.44	84.79	86.14	87.50	85.99	**89.18**
Software	84.69	85.66	87.33	**88.66**	84.70	88.33
Sports	82.23	79.05	82.90	**86.75**	81.09	84.61
Music	85.52	**87.50**	75.00	75.00	**87.50**	**87.50**
Baby	86.54	76.47	88.82	88.23	88.52	**92.94**
Home	87.37	85.29	88.39	89.21	85.19	**90.52**
Books	84.89	83.65	88.94	85.57	88.68	**89.42**
Shoes	86.48	82.40	87.60	87.20	88.13	**89.20**
Automotive	**90.00**	81.06	83.98	88.34	83.15	87.37
Bed	84.02	80.62	87.50	**90.62**	85.09	**90.62**
PC	87.83	85.53	90.89	**91.25**	85.19	90.71
Players	87.96	86.80	90.95	91.87	88.96	**92.48**
Camera	88.70	85.85	89.64	90.40	88.88	**90.65**
Tools	87.12	85.73	**90.98**	89.67	87.62	90.16
Audio	89.37	82.45	88.15	90.78	87.71	**91.22**
Phone	87.54	82.90	**89.31**	88.46	84.73	88.46
Laptop	89.69	87.69	89.23	**93.07**	90.05	92.69
TV	88.76	85.46	91.13	90.64	83.63	**91.87**
Network	**91.45**	84.95	90.65	86.99	83.93	90.24
Office	**88.81**	83.25	86.92	88.53	86.99	88.76

FLN suffers from catastrophic forgetting to some extend as mentioned in Sect. 1, but since sentiment classification tasks are similar, it retains some cross-domain knowledge in feature learning.

Comparing SRK with LSC and EWC. We can see that SRK achieves a much better result than EWC. The main reason is that, when EWC is enforced to protect its parameters learned in the past domains, its regularization may hinder the model to reach a better parameter fitting for the new domain. Unlike EWC, SRK does not impose restrictions for the past tasks, i.e., no explicit optimization terms for the previous domains, which enable it to fit the current task better, with the knowledge still being retained and used. LSC, a naïve Bayes-based lifelong learning approach, achieves competitive results compared with other neural models, which implies the usefulness of the knowledge accumulation and utilization. However, LSC cannot model the contextual relationship due to its conditional independence assumption on features (words). Its knowledge is also restricted to some extent by only considering the word frequency/count from different (past) domains. SRK, on the other hand, does not have those limitations.

Performance on All 20 Domains: The full results of all models on 20 domains is reported in Table 3. We can make the following observations: (1) SRK achieves the highest scores more than half of the time across all domains (11 out of 20) and outperforms other models by a large margin. (2) In other domains (9 out of 20) where SRK does not perform the best, we can see its performance is competitive, with almost all less than 1% compared to the best model. SRK is thus very stable/robust. (3) Although other models can attain the highest scores in some domains, their performances are quite unstable, i.e., none of them can reach the best scores consistently in those domains where SRK is not the best.

4.3 Results in Experimental Setting (2)

In the second experimental setting, we evaluate the performance gain in a continuous learning fashion. The results from the consecutive learning tasks are presented in Fig. 4, where four models are compared, namely, I-RNN, FLN, EWC and SRK. Since 2-FLN has the closest performance to FLN in this setting and LSC does not learn the same type of parameters like other neural models, their results are excluded here. This simplified visualization and also makes our analysis more focused and clearer.

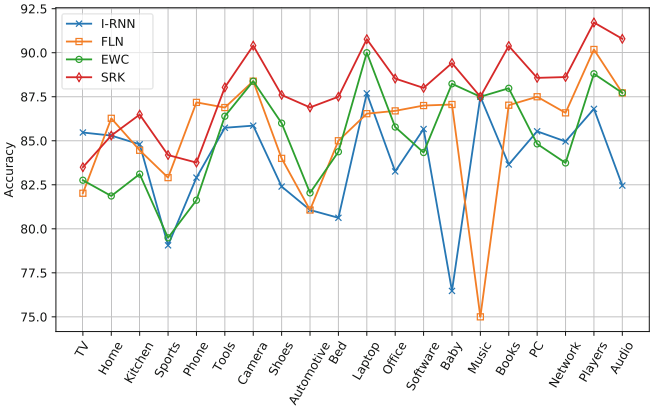

Fig. 4. Accuracy of a specific task sequence.

The conclusion that we can draw from Fig. 4 is three-fold. First, all three continuous learning models (FLN, EWC and SRK) present an overall trend of performance gain over I-RNN from task 1 to task 20, which shows the usefulness of knowledge accumulation during continuous learning. Second, SRK consistently outperforms the other three models. This indicates the superiority of the proposed architecture design in SRK for capturing both domain-specific and cross-domain knowledge. Third, SRK demonstrates its superior stability or robustness to the other models, which we detail next.

Stability/Robustness of SRK. The stability/robustness is a key strength of SRK. Briefly, SRK can address many hard cases where other models can't.

(1) Domain difference. A notable dropping point can be observed from domain *Shoes* to domain *Automotive* in Fig. 4, which can be attributed to their domain differences, i.e., Shoes and Automotive are very different. In this case, the model cannot rely on the parameters learned from its immediate previous (or $(i-1)$ th) domain to improve the current (or i th) domain. In other words, the short-term memorized/residual parameters, which can be viewed as a type of parameter initialization obtained from the last $(i-1)$ task, will not be very helpful for the ith task. This is reflected by both FLN and EWC, where they perform poorly for domain *Automotive*. The proposed SRK is dramatically better in this case because it is able to leverage the knowledge in KRN that is beyond/before the immediate previous domain. However, it is hard for FLN to retain and to exploit such long-term knowledge due to forgetting. EWC mainly aims to deal with catastrophic forgetting by imposing the constraint that its model parameters for learning the i domain are required not to interfere with the old values in the previous domains. It is more likely to ensure the performance of all tasks rather than to learn features or knowledge to help the new task. These explain why SRK reaches a much better result (86.89%) than FLN (81.07%) and EWC (82.04%). A similar situation also occurs from *baby* to *music*.

(2) Imbalanced class distribution in training. In real-world review datasets, the classes are usually imbalanced. Since we used the original class distribution in training, there could be insufficient information to fit the minority class (e.g., the negative class) well and the training/learning might be biased toward the dominating class (e.g., the positive class). This leads to the notably inferior performance for models I-RNN, FLN and EWC for domains *Automotive* and *Shoes*, for models I-RNN and FLN for domains *Baby* and *Music*, respectively. However, this causes no problem for our SRK model as its accumulated knowledge from the past in KRN helps to distinguish positive and negative sentiment signals.

(3) Training data size. When the training data size is small, it causes underfitting. However, since SRK retains the shared knowledge in its KRN (Knowledge Retention Network), this self-learned prior knowledge can provide more supervisory information to help learning to give a better performance. We can see this from domains *Music* and *Bed*, where their dataset sizes are small. In those cases, FLN and I-RNN did not work well but SRK does.

5 Conclusion

In this paper, we proposed a novel deep learning model SRK for lifelong sentiment classification. Specifically, a shared knowledge network is used to retain the knowledge learned from each previous task, and at the same time employed a traditional recurrent neural network to perform feature learning across domains.

Two networks are combined to perform the classification task. For knowledge retention and sharing, a partially update mechanism was designed to preserve the knowledge from individual domains, which has been shown instrumental. Experimental results on real-world datasets shown that the new architecture markedly outperforms the state-of-the-art baselines and is also more stable/robust.

Acknowledgments. This research was partially supported by grants from the National Natural Science Foundation of China (Grants No. 61727809, U1605251) and the program of China Scholarships Council (No. 201706340117). Bing Liu's work was partially supported by National Science Foundation (NSF) under grant nos. IIS1407927 and IIS 1838770, and by Huawei Technologies Co. Ltd with a research gift.

References

1. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In: ACL (2007)
2. Broderick, T., Boyd, N., Wibisono, A., Wilson, A.C., Jordan, M.I.: Streaming variational Bayes. In: NIPS (2013)
3. Caruana, R.: Multitask learning. In: Thrun, S., Pratt, L. (eds.) Learning to Learn. Springer, Boston (1998). https://doi.org/10.1007/978-1-4615-5529-2_5
4. Chen, H., Sun, M., Tu, C., Lin, Y., Liu, Z.: Neural sentiment classification with user and product attention. In: EMNLP (2016)
5. Chen, P., Sun, Z., Bing, L., Yang, W.: Recurrent attention network on memory for aspect sentiment analysis. In: EMNLP (2017)
6. Chen, Z., Liu, B.: Topic modeling using topics from many domains, lifelong learning and big data. In: ICML (2014)
7. Chen, Z., Liu, B.: Lifelong machine learning. Synth. Lect. Artif. Intell. Mach. Learn. **10**, 1–45 (2016)
8. Chen, Z., Ma, N., Liu, B.: Lifelong learning for sentiment classification. In: ACL (2015)
9. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014)
10. French, R.M.: Catastrophic forgetting in connectionist networks. Trends Cogn. Sci. **3**, 128–135 (1999)
11. Ghahramani, Z., Attias, H.: Online variational Bayesian learning. In: Slides from Talk Presented at NIPS Workshop on Online Learning (2000)
12. Glorot, X., Bordes, A., Bengio, Y.: Domain adaptation for large-scale sentiment classification: a deep learning approach. In: ICML (2011)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997)
14. Katiyar, A., Cardie, C.: Investigating LSTMs for joint extraction of opinion entities and relations. In: ACL (2016)
15. Kirkpatrick, J., et al.: Overcoming catastrophic forgetting in neural networks. PNAS **114**, 3521–3526 (2017)
16. Kumaran, D., Hassabis, D., McClelland, J.L.: What learning systems do intelligent agents need? Complementary learning systems theory updated. Trends Cogn. Sci. **20**, 512–534 (2016)

17. Li, X., Lam, W.: Deep multi-task learning for aspect term extraction with memory interaction. In: EMNLP (2017)
18. Li, Z., Zhang, Y., Wei, Y., Wu, Y., Yang, Q.: End-to-end adversarial memory network for cross-domain sentiment classification. In: IJCAI (2017)
19. Liu, B.: Sentiment analysis and opinion mining. Synth. Lect. Hum. Lang. Technol. **5**, 1–67 (2012)
20. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. arXiv preprint arXiv:1502.02791 (2015)
21. Nakagawa, T., Inui, K., Kurohashi, S.: Dependency tree-based sentiment classification using CRFs with hidden variables. In: HLT-NAACL (2010)
22. Pan, S.J., Yang, Q.: A survey on transfer learning. TKDE **22**, 1345–1359 (2010)
23. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Found. Trends® Inf. Retr. **2**, 1–135 (2008)
24. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using machine learning techniques. In: EMNLP (2002)
25. Pennington, J., Socher, R., Manning, C.: GloVe: global vectors for word representation. In: EMNLP (2014)
26. Poria, S., Cambria, E., Gelbukh, A.: Aspect extraction for opinion mining with a deep convolutional neural network. Knowl.-Based Syst. **108**, 42–49 (2016)
27. Qian, Q., Huang, M., Lei, J., Zhu, X.: Linguistically regularized LSTM for sentiment classification. In: ACL (2017)
28. Rozantsev, A., Salzmann, M., Fua, P.: Beyond sharing weights for deep domain adaptation. IEEE Trans. Pattern Anal. Mach. Intell. **41**, 801–814 (2018)
29. Ruvolo, P., Eaton, E.: ELLA: an efficient lifelong learning algorithm. In: ICML (2013)
30. Saha, A., Rai, P., Venkatasubramanian, S., Daume, H.: Online learning of multiple tasks and their relationships. In: AISTATS (2011)
31. Silver, D.L., Yang, Q., Li, L.: Lifelong machine learning systems: beyond learning algorithms. In: AAAI Spring Symposium: Lifelong Machine Learning (2013)
32. Tang, D., Qin, B., Liu, T.: Learning semantic representations of users and products for document level sentiment classification. In: ACL (2015)
33. Tay, Y., Tuan, L.A., Hui, S.C.: Dyadic memory networks for aspect-based sentiment analysis. In: CIKM. ACM (2017)
34. Thrun, S.: Lifelong learning algorithms. In: Thrun, S., Pratt, L. (eds.) Learning to Learn. Springer, Boston (1998). https://doi.org/10.1007/978-1-4615-5529-2_8
35. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep domain confusion: maximizing for domain invariance. arXiv preprint arXiv:1412.3474 (2014)
36. Wang, J., Yu, L.C., Lai, K.R., Zhang, X.: Dimensional sentiment analysis using a regional CNN-LSTM model. In: ACL (2016)
37. Wang, J., et al.: Aspect sentiment classification with both word-level and clause-level attention networks. In: IJCAI (2018)
38. Wang, X., Wei, F., Liu, X., Zhou, M., Zhang, M.: Topic sentiment analysis in Twitter: a graph-based hashtag sentiment classification approach. In: CIKM (2011)
39. Wang, Y., Huang, M., Zhao, L., et al.: Attention-based LSTM for aspect-level sentiment classification. In: EMNLP (2016)
40. Wang, Y., et al.: Dual transfer learning for neural machine translation with marginal distribution regularization. In: AAAI (2018)
41. Xu, J., Chen, D., Qiu, X., Huang, X.: Cached long short-term memory neural networks for document-level sentiment classification. In: EMNLP (2016)
42. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: NAACL (2016)

43. Yessenalina, A., Yue, Y., Cardie, C.: Multi-level structured models for document-level sentiment classification. In: EMNLP (2010)
44. Yu, J., Jiang, J.: Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In: EMNLP (2016)
45. Zhang, J., Ghahramani, Z., Yang, Y.: Flexible latent variable models for multi-task learning. Mach. Learn. **73**, 221–242 (2008)
46. Zhang, K., et al.: Image-enhanced multi-level sentence representation net for natural language inference. In: ICDM, pp. 747–756. IEEE (2018)
47. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: NIPS (2015)
48. Zhou, X., Wan, X., Xiao, J.: Attention-based LSTM network for cross-lingual sentiment classification. In: EMNLP (2016)