Reinforced Co-Training

Jiawei Wu
Department of Computer Science
University of California
Santa Barbara, CA 93106 USA
jiawei_wu@cs.ucsb.edu

Lei Li
Toutiao AI Lab
Bytedance Co. Ltd
Beijing, 100080 China
lileicc@gmail.com

William Yang Wang
Department of Computer Science
University of California
Santa Barbara, CA 93106 USA
william@cs.ucsb.edu

Abstract

Co-training is one popular semi-supervised learning framework to utilize a large amount of unlabeled data in addition to a small labeled set. Co-training methods exploit predicted labels on the unlabeled data and select samples based on prediction confidence to augment the training. However, the sample selection in existing co-training methods is based on a predetermined policy, which ignores the sampling bias between the unlabeled and the labeled, and fails to explore data space. In this paper, we propose a method, Reinforced Co-Training (RCT), to select high-quality unlabeled samples to better co-train on. More specifically, our method uses Q-learning to learn a data selection policy with a small labeled dataset, and then exploits this policy to automatically train the co-training classifiers. Experimental results on click-bait detection and generic text classification tasks demonstrate that our proposed method is able to obtain more accurate text classification.

1 Introduction

Large labeled datasets are required to obtain satisfactory performance for natural language processing tasks. However, it is time-consuming to manually label text corpus. For example, manually attaching a class label for an article usually takes more time than labeling an image. In the meanwhile, there are abundant unlabeled text corpora available on the web. Semi-supervised methods permit learning improved models by jointly train on a small labeled dataset and a large unlabeled set (Zhu, 2006; Chapelle et al., 2009).

Co-training is one of the widely used semi-supervised methods, where two complementary classifiers utilize large amounts of unlabeled examples to bootstrap the performance of each other iteratively (Blum and Mitchell, 1998). Co-training can be readily applied to NLP tasks since data in these tasks naturally have two or more views, such as multi-lingual data (Wan, 2009) and document data (headline and content) (Ghani, 2000; Denis et al., 2003). In the co-training framework, each classifier is trained on one of the two views (aka a subset of features) of both labeled and unlabeled data, under the assumption that either view is sufficient to classify. In each iteration, the co-training algorithm selects high confidence samples predicted and scored by either of the classifiers to form an auto-labeled dataset, and the other classifier is then updated with both labeled data and additional auto-labeled set. However, as shown in Figure 1, most of existing co-training methods have some disadvantages. Firstly, the sample selection step ignores distributional bias between the labeled and unlabeled sets. It is common in practice to use unlabeled datasets collected differently from the labeled set, resulting in a significant difference in their sample distribution. After itera-

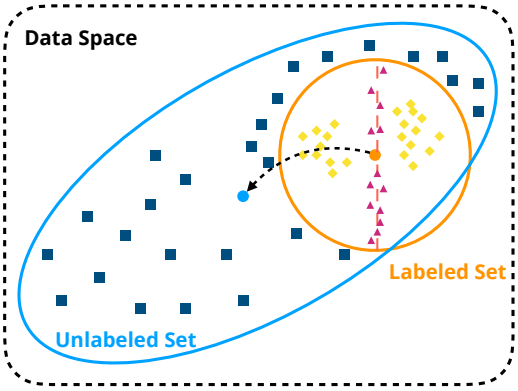

Figure 1: Illustration of sample-selection issues in co-training methods. (1) Randomly sampled unlabeled examples will result in high sampling bias (□), which will cause bias shift towards the unlabeled dataset (←). (2) High-confidence examples (◇) will contribute little during the model training, especially for discriminating the boundary examples (△), resulting in myopic trained models.

tive co-training, the sampling bias may shift towards the unlabeled set, which results in poor performance of the trained model at the testing time. To remedy such bias, an ideal algorithm should select those samples according to the target (potentially unknown) testing distribution. Secondly, the existing sample selection and training can be myopic. Conventional co-training methods select unlabeled examples with high confidence predicted by trained models. This strategy often causes only those unlabeled examples that match well to the current model being picked during iteration and the model might fail to generalize to complete sample space (Zhang and Rudnicky, 2006). It relates to the well-known exploration-exploitation trade-off in machine learning tasks. An ideal co-training algorithm should explore the space well to achieve globally better performance. These intuitions inspire our work on learning a data selection policy for the unlabeled dataset in co-training.

The iterate data selection steps in co-training can be viewed as a sequential decision-making problem. To resolve both issues discuss above, we propose **Reinforced Co-Training**, a reinforcement learning (RL)-based framework for co-training. Concretely, we introduce a joint formulation of a Q-learning agent and two co-training classifiers. In contrast to previous fixed data sampling methods of co-training, we design a Q-agent to automatically learn a data selection policy to select high-quality unlabeled examples. To better guide the Q-agent for policy learning, we design a novel state representation to delivery the status of classifiers and utilize the validation set to compute the performance-driven rewards. Empirically, we show that our method outperforms previous related methods on click-bait detection and generic text classification problems. In summary, our main contributions are three-fold:

- We are first to propose a joint formulation of RL and co-training methods;
- Our learning algorithm can learn a good data selection policy to select high-quality unlabeled examples for better co-training;
- We show that our method can apply to large-scale document data and outperform baselines in semi-supervised text classification.

In Section 2, we outline related work in semi-supervised learning and co-training. We then describe our proposed method in Section 3. We show experimental results in Section 4. Finally, we conclude in Section 5.

2 Related Work

Semi-supervised learning algorithms have been widely used in NLP (Liang, 2005). As for text classification, Dai and Le (2015) introduce a sequence autoencoder to pre-train the parameters for the later supervised learning process. Johnson and Zhang (2015, 2016) propose a method to learn embeddings of small text regions from unlabeled data for integration into a supervised convolutional neural network (CNN) or long short-term memory network (LSTM). Miyato et al. (2016) further apply perturbations to the word embeddings and pre-train the supervised models through adversarial training. However, these methods mainly focus on learning the local word-level information and pre-trained parameters from unlabeled data, which fails to capture the overall text-level information and potential label information.

Co-training can capture the text-level information of unlabeled data and generate pseudo labels during the training, which is especially effective on unlabeled data with two distinct views (Blum and Mitchell, 1998). However, the confidence-based data selection strategies (Goldman and Zhou, 2000; Zhou and Li, 2005; Zhang and Zhou, 2011) often focus on some special regions of the input space and fail to generate an accurate estimation of data space. Zhang and Rudnicky (2006) proposes a performance-driven data selection strategy based on pseudo-accuracy and energy regularization. Meanwhile, Chawla et al. (2005) argues that the random data sampling method often causes sampling bias shift of the trained model towards the unlabeled set.

Comparing to previous related methods, our Reinforced Co-Training model can learn a performance-driven data selection policy to select high-quality unlabeled data. Furthermore, the performance estimation is more accurate due to the validation dataset and the data selection strategy is automatically learned instead of human designed. Lastly, the selected high-quality unlabeled data can not only help explore the data space but also reduce the sampling bias shift.

3 Method

In this section, we describe our RL-based framework for co-training in detail. The conventional

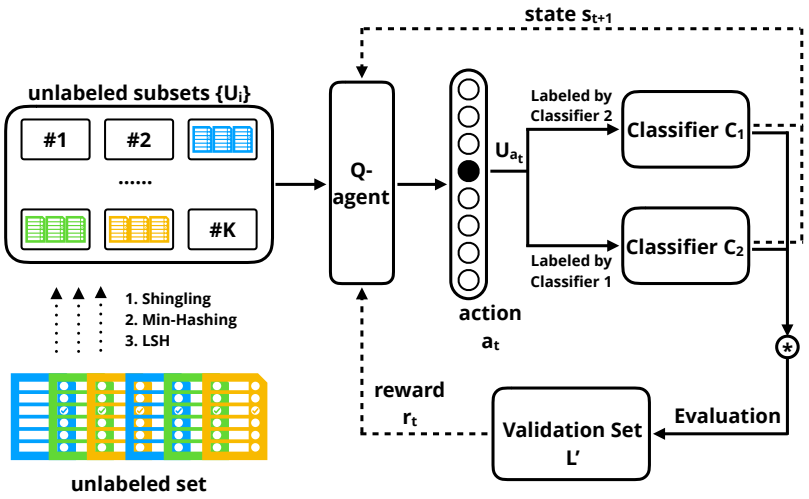

Figure 2: The Reinforced Co-Training framework.

co-training methods follow the framework: 1) initialize two classifiers by training on the labeled set, and iteratively 2) select a subset of unlabeled data based on a predefined policy, and 3) update two classifiers with the selected subset of unlabeled data in addition to the labeled one. Step 2 is the core of different co-training variants. The original co-training algorithm is equipped with a policy of selecting high-confidence samples by two classifiers. Our main idea is to improve the policy by reinforcement learning.

We formulate the data selection process as a sequential decision-making problem and the decision (action) a_t at each iteration (time step) t is to select a portion of unlabeled examples. This problem can be solved with an RL-agent by learning a policy. We first describe how we organize the large unlabeled dataset to improve the computational efficiency. Then we briefly introduce the classifier models used in co-training. After that, we describe the Q-agent, the RL-agent used in our framework and the environment in RL. The two co-training classifiers are integrated into the environment and the Q-agent can learn a good data selection policy by interacting with the environment. Finally, we describe how to train the Q-agent in our unified framework.

3.1 Partition Unlabeled Data

Considering that the number of unlabeled samples is enormous, it is not efficient for the RL-agent to select only one example at each time step t. Thus, first we want to partition documents from the unlabeled dataset into different subsets based on their similarity. At each time step t, the RL-agent applies a policy to select one subset instead of one sample and then update the two co-training classifiers, which can significantly improve the computational efficiency.

Suppose each example in the unlabeled dataset as document D, where D is the concatenation of the headline and paragraph. V is the vocabulary of these documents. These documents are partitioned into different subsets based on Jaccard similarity, which is defined as: $sim(D_1, D_2) = \frac{|D_1 \cap D_2|}{|D_1 \cup D_2|}$, where $D_1, D_2 \in \mathbb{R}^{|V|}$ are the one-hot vectors of each document example.

Based on Jaccard similarity, the unlabeled examples can be spit into different subsets using the following three steps, which have been widely used in large-scale web search (Rajaraman and Ullman, 2010): 1) Shingling, 2) Min-Hashing, and 3) Locality-Sensitive Hashing (LSH).

After partition, the unlabeled set U can be converted into K different subset $\{U_1, U_2, ..., U_K\}$. Meanwhile, for each subset U_i, the first added document example S_i is recorded as the representative example of the subset U_i.

3.2 Classifier Models

As mentioned before, many linguistic data naturally has two or more views, such as multi-lingual data (Wan, 2009) and document data (headline + paragraph) (Ghani, 2000; Denis et al., 2003). Based on the two views of data, we can construct two classifiers respectively. At the beginning of a training episode, the two classifiers are first seeded

with a small set of labeled (seeding) training data L. At each time step t, the RL-agent makes a selection action a_t, and then the unlabeled subset U_{a_t} is selected to train the two co-training classifiers. Following the standard co-training process (Blum and Mitchell, 1998), at each time step t, the classifier C_1 annotate the unlabeled subset U_{a_t} and the pseudo-labeled U_{a_t} is then used to update the classifier C_2, vice versa. In this way, we can boost the performance of C_1 and C_2 simultaneously.

3.3 Q-learning Agent

Q-learning is a widely used method to find an optimal action-selection policy (Watkins and Dayan, 1992). The core of our model is a Q-learning agent, which is trained to learn a good policy to select high-quality unlabeled subsets for co-training. At each time step t, the agent observes the current state s_t, and selects an action a_t from a discrete set of actions $A = \{1, 2, ..., K\}$. Based on the action a_t, the two co-training classifiers C_1 and C_2 then can be updated with the unlabeled subset U_{a_t} as described in Section 3.2. After that, the agent receives a performance-driven reward r_t and the next state observation s_{t+1}. The agent's goal at each time step t is to choose the action that maximizes the future discount reward

$$R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}, \tag{1}$$

where a training episode terminates at time T and γ is the discount factor.

3.3.1 State Representation

The state representation, in our framework, is designed to deliver the status of two co-training classifiers to the Q-agent. Zhang and Rudnicky (2006) have proved that training with high-confidence examples will consequently be a process that reinforces what the current model already encodes instead of learning an accurate distribution of data space. Thus, one insight in formulating the state representation is to add some unlabeled examples with diversity and uncertainty during the training iteration. However, too much diversity and uncertainty will also cause the sampling bias shift towards the unlabeled dataset (Yeung et al., 2017). In order to reason about this, the Q-agent needs to fully understand the distribution of the unlabeled data.

Based on the above intuition, we formulate the agents state using the two classifiers' probability distribution on the representative example S_i of each unlabeled subset U_i. Suppose a N-class classification problem, at each time step t, we evaluate the probability distribution of two classifiers on S_i separately. The state representation then can be defined as:

$$s_t = \{P_1^1||P_1^2, P_2^1||P_2^2, ..., P_K^1||P_K^2\}_t, \tag{2}$$

where P_i^1 and P_i^2 are the probability distribution of C_1 and C_2 on S_i separately, and $||$ denotes the concatenation operation. $P_i^1, P_i^2 \in \mathbb{R}^N$ and $P_i^1||P_i^2 \in \mathbb{R}^{2N}$. Note that the state representation is re-computed at each time step t.

3.3.2 Q-network

The agent takes an action at at time step t using a policy

$$a_t = \max_a Q(s_t, a), \tag{3}$$

where s_t is the state representation mentioned above. The Q-value $Q(s_t, a)$ is determined by a neural network as illustrated in Figure 3. Concretely,

$$z_a = \phi(\{F(P_1^1||P_1^2), ..., F(P_K^1||P_K^2)\}; \theta), \tag{4}$$

where the function F maps state representation $P_i^1||P_i^2 \in \mathbb{R}^{2N}$ into a common embedding space of y dimensions, and $\phi(\cdot)$ is a multi-layer perception.

We then use

$$Q(s, a) = \text{softmax}(z_a; \tau) \tag{5}$$

to obtain the next action, where τ is a temperature parameter.

3.3.3 Reward Function

The agent is trained to select the high-quality unlabeled subsets to improve the performance of the two classifier C_1 and C_2. We capture this intuition by a performance-driven reward function. At time step t, the reward of each classifier is defined as the change in the classifiers accuracy after updating the unlabeled subset U_t:

$$r_t^1 = \text{Acc}_t^1(L') - \text{Acc}_{t-1}^1(L'), \tag{6}$$

where $\text{Acc}_t^1(L')$ is the model accuracy of C_1 at time step t computed on the labeled validation set L'. Then the r_t^2 is defined following the similar formulation. The final reward r_t is defined as:

$$r_t = \begin{cases} r_t^1 \times r_t^2 & \text{if } r_t^1 \times r_t^2 > 0, \\ 0 & \text{if } r_t^1 \times r_t^2 \leq 0. \end{cases}$$

Note that this reward is only available during training process.

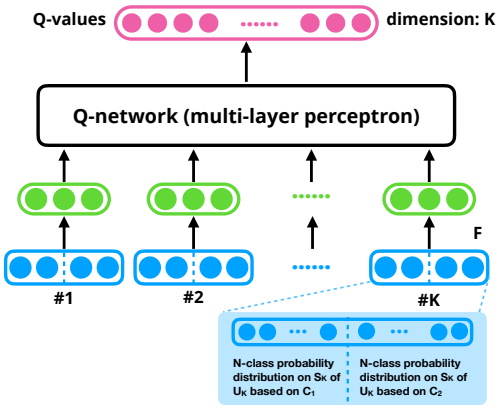

Figure 3: The structure of Q-network. It chooses a unlabeled subset from $\{U_1, U_2, ..., U_K\}$ at each time step. The state representation is computed according to the two classifiers' N-class probability distribution on the representative example S_i of each subset U_i.

3.4 Training and Testing

The agent is trained with the Q-learning (Watkins and Dayan, 1992), a standard reinforcement learning algorithm that can be used to learn policies for an agent interacting with an environment. In our Reinforced Co-Training framework, the environment is the classifier C_1 and C_2.

The Q-network parameters θ are learned by optimizing:

$$L_i(\theta_i) = \mathbb{E}_{s,a}[(V(\theta_{i-1}) - Q(s, a; \theta_i))^2], \quad (7)$$

where i is an iteration of optimization and

$$V(\theta_{i-1}) = \mathbb{E}_{s'}[r + \gamma \max_{a'} Q(s', a'; \theta_{i-1})|s, a]. \quad (8)$$

.

We optimize it using stochastic gradient descent. The detail of the training process is shown in Algorithm 1.

At test time, the agent and the two co-training classifiers are again run simultaneously, but without access to the labeled validation dataset. The agent selects the unlabeled subset using the learned greedy policy: $a_t = max_a Q(s_t, a)$. After obtaining two classifiers from co-training, based on the weighted voting, the final ensemble classifier C is defined as:

$$C = \beta C_1 + (1 - \beta)C_2. \quad (9)$$

β is the weighted parameter, which can be learned by maximizing the classification accuracy on the validation set.

Algorithm 1: The algorithm of our Reinforced Co-Training method.

Given a set L of labeled seeding training data;
Given a set L' of labeled validation data;
Given K subsets $\{U_1, U_2, ..., U_K\}$ of unlabeled data;
for *episode* $\leftarrow$ *1* **to** *M* **do**
  Train C_1 & C_2 with L
  for *time step* $t \leftarrow$ *1* **to** T **do**
    Choose the action $a_t = \max_a Q(s_t, a)$
    Use C_1 to label the subset U_{a_t}
    Update C_2 with pseudo-labeled U_{a_t}
    Use C_2 to label the subset U_{a_t}
    Update C_1 with pseudo-labeled U_{a_t}
    Compute the reward r_t based on L'
    Compute the state representation s_{t+1}
  Update θ using $g \propto \nabla_\theta \mathbb{E}_{s,a}[(V(\theta_{i-1}) - Q(s, a; \theta_i))^2]$

4 Experiments

We evaluate our proposed Reinforced Co-training method in two settings: (2) **Clickbait detection**, where obtaining the labeled data is very time-consuming and labor-intensive in this real-world problem; (1) **Generic text classification**, where we randomly set some of the labeled data as unlabeled and train our model in a controlled setting.

4.1 Baselines

We compare our model with multiple baselines:

- **Standard Co-Training**: Co-Training with randomly choosing unlabeled examples (Blum and Mitchell, 1998).
- **Performance-driven Co-Training**: The unlabeled examples are selected based on pseudo-accuracy and energy regularization (Zhang and Rudnicky, 2006).
- **CoTrade Co-Training**: The confidence of either classifiers prediction on unlabeled examples is estimated based on specific data editing techniques, and then high-confidence examples are used to update the classifiers (Zhang and Zhou, 2011).
- **Semi-supervised Sequence Learning (Sequence-SSL)**: The model uses an LSTM sequence autoencoder to pre-train the parameters for the later supervised learning process.(Dai and Le, 2015).

- **Semi-supervised CNN with Region Embedding (Region-SSL)**: The model learns embeddings of small text regions from unlabeled data for integration into a supervised CNN (Johnson and Zhang, 2015).
- **Adversarial Semi-supervised Learning (Adversarial-SSL)**: The model apply perturbations to word embeddings into an LSTM and pre-train the supervised models through adversarial training (Miyato et al., 2016).

4.2 Clickbait Detection

Clickbait is a pejorative term for web content whose headlines typically aim to make readers curious, but the documents usually have less relevance with the corresponding headlines (Chakraborty et al., 2016; Potthast et al., 2017; Wei and Wan, 2017). Clickbait not only wastes the readers' time but also damages the publishers' reputation, which makes detecting clickbait become an important real-world problem.

However, most of the attempts focus on news headlines, while the relevance between headlines and context is usually ignored (Chen et al., 2015; Biyani et al., 2016; Chakraborty et al., 2016). Meanwhile, the labeled data is quite limited in this problem, but the unlabeled data is easily obtained from the web (Potthast et al., 2017). Considering these two challenges, we utilize our Reinforced Co-training framework to tackle this problem and evaluate our method.

4.2.1 Datasets

We evaluate our model on a large-size clickbait dataset, Clickbait Challenge 2017 (Potthast et al., 2017). The data is collected from twitter posts including tweet headlines and paragraphs, and the training and test sets are judged on a four-point scale $[0, 0.3, 0.66, 1]$ by at least five annotators. Each sample is categorized into one class based on its average scores. The clickbait detection then can be defined as a two-class classification problem, including CLICKBAIT and NON-CLICKBAIT. There also exists an unlabeled set containing large amounts of collected samples without annotation. We then split the original test set into the validation set and final test set by 50%/50%. The statistics of this dataset are listed in Table 1.

Dataset	#Tweets	#Clickbait	#Non-Clickbait
Training	2,495	762	1,697
Validation	9,768	2,380	7,388
Test	9,770	2,381	7,389
Unlabeled	80,012	N/A	N/A

Table 1: Statistics of Clickbait Dataset.

4.2.2 Setup

For each document example in the clickbait dataset, naturally, we have two views, the headline and the paragraph. Thus, we construct the two classifiers in co-training based on these two views.

Headline Classifier The previous state-of-the-art model (Zhou, 2017) for clickbait detection uses a self-attentive bi-directional gated recurrent unit RNN (biGRU) to model the headlines of the document and train a classifier. Following the same setting, we choose self-attentive biGRU as the headline classifier in co-training.

Paragraph Classifier The paragraphs usually have much longer sequences than the headlines. Thus, we utilize a CNN structure in Kim (2014) as the paragraph classifier to capture the paragraph information.

Note that the other three co-training baselines also use the same classifier settings.

In our Reinforce Co-Training model, we set the number of unlabeled subsets k as 80. Considering the clickbait detection as a 2-class classification problem ($N = 2$), the Q-network maps 4-d input $P_i^1||P_i^2$ in the state representation to a 3-d common embedding space ($y = 3$), with a further hidden layer of 128 units on top. The dimension k of the softmax layer is also 80. A temperature parameter of $\tau = 60$ is used in the Q-network. The training process consists of $M = 300$ episodes, and each episode terminates at time $T = 200$.

As for the other semi-supervised baselines, Sequence-SSL, Region-SSL and Adversarial-SSL, we concatenate the headline and the paragraph as the document and train these models directly on the document data. To better analyze the experimental results, we also implement another baseline denoted as CNN (Document), which uses the CNN structure (Kim, 2014) to model the document with supervised learning. The CNN (Document) model is trained on the (seeding) training set and the validation set.

Following the previous researches (Chakraborty et al., 2016; Potthast et al., 2017), we use Precision, Recall and F1 Score to evaluate different models.

Methods	Prec.	Recall	F1 Score
Self-attentive biGRU	0.683	0.649	0.665
CNN (Document)	0.537	0.474	0.503
Standard Co-Training	0.418	0.433	0.425
Performance Co-Training	0.581	0.629	0.604
CoTrade Co-Training	0.609	0.637	0.623
Sequence-SSL	0.595	0.589	0.592
Region-SSL	0.674	0.652	0.663
Adversarial-SSL	0.698	**0.691**	0.694
Reinforced Co-Training	**0.709**	0.684	**0.696**

Table 2: The experimental results on clickbait dataset. Prec.: precision.

4.2.3 Results

The results of clickbait detection are shown in Table 2. From the results, we observe that: (1) Our Reinforced Co-Training model can outperform all the baselines, which indicates the capability of our methods in utilizing the unlabeled data. (2) The standard co-training is unstable due to the random data selection strategy, and the performance-driven and high-confidence data selection strategies both can improve the performance of co-training. Meanwhile, the significant improvement compared with previous co-training methods shows that the Q-agent in our model can learn a good policy to select high-quality subsets. (3) The three pre-trained based semi-supervised learning methods also show good results. We think these pre-trained based methods learn local embeddings during the unsupervised training, which may help them to recognize some important patterns in clickbait detection. (4) The self-attentive biGRU trained only on headlines of the labeled set actually show surprisingly good performance on clickbait detection, which demonstrates that most clickbait documents have obvious patterns in the headline field. The reason why CNN (Document) fails to capture these patterns may be that the concatenation of headlines and paragraphs dilutes these features. But for those cases without obvious patterns in the headline, our results demonstrate that the paragraph information is still a good supplement to detection.

4.2.4 Algorithm Robustness

Previous researches (Morimoto and Doya, 2005; Henderson et al., 2017) show that reinforcement learning-based methods usually lack robustness and are sensitive to the seeding sets and pre-trained steps. Thus, we design an experiment to detect whether our learned data section policy is sensitive to the (seeding) training set. First, based on our original data partition, we train our reinforcement learning framework to learn a Q-agent. During the test time, instead of using the same seeding set, we randomly sample other 10 seeding sets from the labeled dataset and learn 10 classifiers based on this learned Q-agent. Note that the validation set is not available during the co-training period of the test time. Finally, we evaluate these 10 classifiers using the same metric. The results are shown in Figure 3.

	Best	Worst	Average	STDDEV
F1 Score	0.708	0.685	0.692	0.0068

Table 3: The robustness analysis on clickbait dataset.

The results demonstrate that our learning algorithm is robust to different (seeding) training sets, which indicates that the Q-agent in our model can learn a good and robust data selection policy to select high-quality unlabeled subsets to help the co-training process.

4.3 Generic Text Classification

Generic text classification is a classic problem for natural language processing, where one needs to categorized documents into pre-defined classes (Kim, 2014; Zhang et al., 2015; Johnson and Zhang, 2015, 2016; Xiao and Cho, 2016; Miyato et al., 2016). We evaluate our model on generic text classification problem to study our method in a controlled setting.

4.3.1 Datasets

Following the settings in Zhang et al. (2015), we use large-scale datasets to train and test our model. To maintain the two-view setting of the co-training method, we choose the following two datasets. The original annotated training set is then split into three sets, 10% labeled training set, 10% labeled validation set and 80% unlabeled set. The original proportion of different classes remains the same after the partition. The statistics of these two datasets are listed in Table 4.

AG's news corpus. The AGs corpus of news articles is obtained from the web and each sample has the title and description fields.

DBPedia ontology dataset. This dataset is constructed by picking 14 non-overlapping classes from DBpedia 2014. Each sample contains the title and abstract of a Wikipedia article.

Dataset	AG's News	DBPedia
#Classes	4	14
#Training	12,000	56,000
#Validation	12,000	56,000
#Test	7,600	70,000
#Unlabeled	96,000	448,000

Table 4: Statistics of the Text Classification Datasets.

4.3.2 Setup

For each document example in the above two datasets, naturally we have two views, the headline and the paragraph. Similar to clickbait detection, we also construct the two classifiers in co-training based on these two views. Following the (Kim, 2014), we set both the headline classifier and the paragraph classifier as the CNN model. Owing to that fact that the original datasets are fully labeled, we implement two other baselines: (1) CNN (Training+Validation), which is supervised trained on the partitioned training and validation sets; (2) CNN (All) which is supervised trained on the original (100%) dataset.

For AG's News dataset, we set the number of unlabeled subsets k as 96. The number of classes $N = 4$, and thus the Q-network maps 8-d input $P_i^1||P_i^2$ in the state representation to a 5-d common embedding space ($y = 5$), with a further hidden layer of 128 units on top. The dimension k of the softmax layer is also 96. A temperature parameter of $\tau = 80$ is used in the Q-network. The training process consists of $M = 400$ episodes, and each episode terminates at time $T = 250$. As for DBPedia dataset, $k = 224, N = 14, \tau = 150$, $y = 10$, $M = 800$ and $T = 400$.

Following the previous researches (Kim, 2014), we use test error rate (%) to evaluate different models.

4.3.3 Results

The results of generic text classification are shown in Table 5. From the results, we can observe that: (1) Our Reinforced Co-Training model outperforms all the real semi-supervised baselines on two generic text classification datasets, which indicates that our method is consistent in different tasks. (2) The CNN (All) and Adversarial-SSL trained on all the original labeled data perform best, which indicates there is still an obvious gap between semi-supervised methods and full-supervised methods.

Methods	**AG's News**	**DBPedia**
CNN (Training+Validation)	28.32%	9.53%
CNN (All)	8.69%	0.91%
Standard Co-Training	26.52%	7.66%
Performance Co-Training	21.73%	5.84%
CoTrade Co-Training	19.06%	5.12%
Sequence-SSL	19.54%	4.64%
Region-SSL	18.27%	3.76%
Adversarial-SSL	8.45%*	0.89%*
Reinforced Co-Training	**16.64%**	**2.45%**

Table 5: The experimental results on generic text classification datasets. * Adversarial-SSL is trained on full labeled data after pre-training.

Datasets	**Best**	**Worst**	**Average**	**STDDEV**
AG's News	14.78	17.96	16.62	1.36
DBPedia	2.18	4.06	2.75	0.94

Table 6: The robustness analysis on generic text classification. Metric: test error rate (%).

4.3.4 Algorithm Robustness

Similar to Section 4.2.4, we evaluate whether our learned data section policy is sensitive to the different partitions and (seeding) training sets. First, based on our original data partition (10%/10%/80%), we train our reinforcement learning framework. During the test time, we randomly sample other 10 data partitions and learn 10 ensemble classifiers based on the learned Q-agent. Note that after sample different data partitions, we will also reprocess the unlabeled sets as described in Section 3.1. We then evaluate these 10 classifiers using the same metric. The results are shown in Table 6.

The results demonstrate that our learning algorithm is robust to different (seeding) training sets and partitions of the unlabeled set, which again indicates that the Q-agent in our model is able to learn a good and robust data selection policy to select high-quality unlabeled subsets to help the co-training process.

5 Conclusion

In this paper, we propose a novel method, Reinforced Co-Training, for training classifiers by utilizing both the labeled and unlabeled data. The Q-agent in our model can learn a good data selection policy to select high-quality unlabeled data for co-training. We evaluate our models on two tasks, click-bait detection and generic text classification. Experimental results show that our model outperforms other semi-supervised baselines, especially those conventional co-training methods.

References

Prakhar Biyani, Kostas Tsioutsiouliklis, and John Blackmer. 2016. " 8 amazing secrets for getting more clicks": Detecting clickbaits in news streams using article informality. In *AAAI*. pages 94–100.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*. ACM, pages 92–100.

Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. 2016. Stop clickbait: Detecting and preventing clickbaits in online news media. In *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*. IEEE, pages 9–16.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. 2009. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks* 20(3):542–542.

Nitesh V Chawla et al. 2005. Learning from labeled and unlabeled data: An empirical study across techniques and domains. *Journal of Artificial Intelligence Research* 23:331–366.

Yimin Chen, Niall J Conroy, and Victoria L Rubin. 2015. Misleading online content: Recognizing clickbait as false news. In *Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection*. ACM, pages 15–19.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*. pages 3079–3087.

Francois Denis, Anne Laurent, Rémi Gilleron, and Marc Tommasi. 2003. Text classification and co-training from positive and unlabeled examples.

Rayid Ghani. 2000. Using error-correcting codes for text classification. In *ICML*. pages 303–310.

Sally Goldman and Yan Zhou. 2000. Enhancing supervised learning with unlabeled data. In *ICML*. pages 327–334.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. 2017. Deep reinforcement learning that matters. *arXiv preprint arXiv:1709.06560* .

Rie Johnson and Tong Zhang. 2015. Semi-supervised convolutional neural networks for text categorization via region embedding. In *Advances in neural information processing systems*. pages 919–927.

Rie Johnson and Tong Zhang. 2016. Supervised and semi-supervised text categorization using lstm for region embeddings. In *International Conference on Machine Learning*. pages 526–534.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* .

Percy Liang. 2005. *Semi-supervised learning for natural language*. Ph.D. thesis, Massachusetts Institute of Technology.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725* .

Jun Morimoto and Kenji Doya. 2005. Robust reinforcement learning. *Neural computation* 17(2):335–359.

M Potthast, T Gollub, M Hagen, and B Stein. 2017. The clickbait challenge 2017: Towards a regression model for clickbait strength. *Proceddings of the Clickbait Chhallenge* .

A Rajaraman and JD Ullman. 2010. Finding similar items. *Mining of Massive Datasets* 77:73–80.

Xiaojun Wan. 2009. Co-training for cross-lingual sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-volume 1*. Association for Computational Linguistics, pages 235–243.

Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8(3-4):279–292.

Wei Wei and Xiaojun Wan. 2017. Learning to identify ambiguous and misleading news headlines. *arXiv preprint arXiv:1705.06031* .

Yijun Xiao and Kyunghyun Cho. 2016. Efficient character-level document classification by combining convolution and recurrent layers. *arXiv preprint arXiv:1602.00367* .

Serena Yeung, Vignesh Ramanathan, Olga Russakovsky, Liyue Shen, Greg Mori, and Li Fei-Fei. 2017. Learning to learn from noisy web videos. *arXiv preprint arXiv:1706.02884* .

Min-Ling Zhang and Zhi-Hua Zhou. 2011. Cotrade: confident co-training with data editing. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 41(6):1612–1626.

Rong Zhang and Alexander I Rudnicky. 2006. A new data selection principle for semi-supervised incremental learning. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*. IEEE, volume 2, pages 780–783.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*. pages 649–657.

Yiwei Zhou. 2017. Clickbait detection in tweets using self-attentive network. *arXiv preprint arXiv:1710.05364* .

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on knowledge and Data Engineering* 17(11):1529–1541.

Xiaojin Zhu. 2006. Semi-supervised learning literature survey. *Computer Science, University of Wisconsin-Madison* 2(3):4.