
DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS

*STOCKHOLM, SWEDEN 2017*

# Deep Active Learning for Short-Text Classification

**WENQUAN ZHAO**

**KTH ROYAL INSTITUTE OF TECHNOLOGY**
**SCHOOL OF COMPUTER SCIENCE AND COMMUNICATION**

# Deep Active Learning for Short-Text Classification

WENQUAN ZHAO

Master in Computer Science
Date: July 9, 2017
Supervisor: Zhenyu Yang, Erik Fransén
Examiner: Olov Engwall
Swedish title: Aktiv inlärning i djupa nätverk för klassificering av korta texter
School of Computer Science and Communication

## Abstract

In this paper, we propose a novel active learning algorithm for short-text (Chinese) classification applied to a deep learning architecture. This topic thus belongs to a cross research area between acitve learning and deep learning. One of the bottlenecks of deep learning for classification is that it relies on large number of labeled samples, which is expensive and time consuming to obtain. Active learning aims to overcome this disadvantage through asking the most useful queries in the form of unlabeled samples to be labeled. In other words, active learning intends to achieve precise classification accuracy using as few labeled samples as possible. Such ideas have been investigated in conventional machine learning algorithms, such as support vector machine (SVM) for image classification, and in deep neural networks, including convolutional neural networks (CNN) and deep belief networks (DBN) for image classification. Yet the research on combining active learning with recurrent neural networks (RNNs) for short-text classification is rare. We demonstrate results for short-text classification on datasets from Zhuiyi Inc. Importantly, to achieve better classification accuracy with less computational overhead, the proposed algorithm shows large reductions in the number of labeled training samples compared to random sampling. Moreover, the proposed algorithm is a little bit better than the conventional sampling method, uncertainty sampling. The proposed acitve learning algorithm dramatically decreases the amount of labeled samples without significantly influencing the test classification accuracy of the original RNNs classifier, trained on the whole data set. In some cases, the proposed algorithm even achieves better classification accuracy than the original RNNs classifier.

## Sammanfattning

I detta arbete studerar vi en ny aktiv inlärningsalgoritm som appliceras på en djup inlärningsarkitektur för klassificering av korta (kinesiska) texter. Ämnesområdet hör därmed till ett ämnesöverskridande område mellan aktiv inlärning och inlärning i djupa nätverk. En av flaskhalsarna i djupa nätverk när de används för klassificering är att de beror av tillgången på många klassificerade datapunkter. Dessa är dyra och tidskrävande att skapa. Aktiv inlärning syftar till att överkomma denna typ av nackdel genom att generera frågor rörande de mest informativa oklassade datapunkterna och få dessa klassificerade. Aktiv inlärning syftar med andra ord till att uppnå bästa klassificeringsprestanda med användandet av så få klassificerade datapunkter som möjligt. Denna idé har studerats inom konventionell maskininlärning, som tex supportvektormaskinen (SVM) för bildklassificering samt inom djupa neuronnätverk inkluderande bl.a. convolutional networks (CNN) och djupa beliefnetworks (DBN) för bildklassificering. Emellertid är kombinationen av aktiv inlärning och rekurrenta nätverk (RNNs) för klassificering av korta texter sällsynt. Vi demonstrerar här resultat för klassificering av korta texter ur en databas från Zhuiyi Inc. Att notera är att för att uppnå bättre klassificeringsnoggranhet med lägre beräkningsarbete (overhead) så uppvisar den föreslagna algoritmen stora minskningar i det antal klassificerade träningspunkter som behövs jämfört med användandet av slumpvisa datapunkter. Vidare, den föreslagna algoritmen är något bättre än den konventionella urvalsmetoden, osäkherhetsurval (uncertanty sampling). Den föreslagna aktiva inlärningsalgoritmen minska dramatiskt den mängd klassificerade datapunkter utan att signifikant påverka klassificeringsnoggranheten hos den ursprungliga RNN-klassificeraren när den tränats på hela datamängden. För några fall uppnår den föreslagna algoritmen t.o.m. bättre klassificeringsnoggranhet än denna ursprungliga RNN-klassificerare.

# Contents

**Contents** iii

**1 Introduction** 1
- 1.1 Background . . . 1
- 1.2 Research Question and Objective . . . 2
- 1.3 Related Work . . . 3

**2 Methods** 4
- 2.1 Pool-Based Active Learning . . . 4
- 2.2 Criteria for Sampling . . . 5
  - 2.2.1 Random Sampling . . . 6
  - 2.2.2 Uncertainty Sampling . . . 6
  - 2.2.3 Density-Weighted Method . . . 7
  - 2.2.4 Weighted Uncertainty Sampling . . . 8
- 2.3 RNNs with GRUs . . . 10
- 2.4 Cross Validation . . . 12

**3 Experiments and Results** 15
- 3.1 RNNs model . . . 16
- 3.2 Active RNNs model with Random Sampling . . . 19
- 3.3 Active RNNs model with Uncertainty Sampling . . . 21
- 3.4 Active RNNs model with Weighted Uncertainty Sampling . . . 22

**4 Analyses and Conclusion** 24
- 4.1 Analyses . . . 25
- 4.2 Conclusion . . . 28

**5 Future Work** 29
- 5.1 The Distribution of Samples . . . 29
- 5.2 The Error Reduction of Labels . . . 30

**Bibliography** 31

**A Appended Material** 34
- A.1 List of the resources . . . 34
- A.2 Parameters Tables . . . 34

# Chapter 1

# Introduction

In this project, we formulate the research question from an industry problem, text classification task in a company. The text classification problem has been studied for several decades, so the basic ideas to solve this problem are clear. However, there exists a bottleneck for this task under deep learning circumstance, i.e., gathering the labels of samples manually is expensive and time-costing. Active learnig is motivated for overcoming the bottleneck. Actually, active learning is a recurent process for selecting useful samples as training data. The size of training data set should be as small as possible meanwhile keeping the classification performance. As a result, the kernel of active learning is how to select useful samples. In conclusion, we combine active learning and deep learning for text classification task.

In this chapter, section 1.1 introduces the backgroud of the thesis from the points of views fo both industry and academia. Section 1.2 defines the research question based on the point of view of scientific research and describes the objectives according to the company and thesis work. Moreover, section 1.4 presents the related work in active learning area.

## 1.1 Background

Active learning aims to achieve better accuracy with fewer labeled training data through an iterative procedure of selecting the most informative samples based on some criteria. It proves to be efficient in decreasing the amount of training samplings used in several different aspects, such as text classification [6], image classification [1], speech recognition [7] and so on. However, there are three different application scenarios [8] when selecting samples, including: (1) membership query synthesis, in which the learner (trained model) can select arbitrary samples from the whole input space; (2) stream-based selective sampling, in which the learner can decide whether or not to request the label of a sample based on its informativeness from continuous data source; (3) pool-based sampling, in which the learner chooses the best instances out of a fixed large pool of unlabeled data. Active learning with pool-based sampling can be defined as "pool-based active learnig". Our project is in the $3rd$ application scenario. Moreover, the key process of active learning is the criteria for sampling, and several criteria have been proposed. We will select one mature active learning algorithm and apply it in this project. Furthermore, based on the deep learning architecture, we propose a novel active learning algorithm inspired by Settles and Craven [9].

Currently, text classification is one of the key problems of most artificial intelligence companies, which aim to build up chatbot systems or intelligent custom service systems. Recurrent Neural Networks (RNNs) are universial models in many Natural Language Processes (NLP) tasks, such as text classification, speech recognition, and have great performance [4]. However, in order to obtain high classification accuracy based on the RNNs architecture, large number of samples have to be labeled manually, which is expensive and time-consuming. Thus, the low efficient human labeling work becomes the bottleneck of producting good performance of text classification system of these companies.

Actually, both industry and academia motivate reducing the required labeled data in deep learning. The company named Zhuiyi Inc., a representative in industry, shows its interest in the work, applying active learning in short-text classification under the architecture of RNNs. In support of Zhuiyi Inc., I do the thesis project, combining active learning with deep learning. Besides, in an article by Liu et al. [3], the authors proposed that limited training samples restrict the application of deep network in hyperspectral image classification. As a result, active learning is needed both for industry, and for some research domains which have few already labeled data for training a deep neural network.

If active learning could dramatically decrease the amount of labeled training data samples without significantly influencing the accuracy of classification result in text classification application, it will greatly benefit the business of such artificial intelligence companies. In most cases, deep learning requires large amount of training data. Therefore, if active learning performs well combined with RNNs, it will make the deep learning easier to apply from the point of view of gathering and applying small training data set.

To conclude, combining active learning with deep learning has research value on both industry and academia.

## 1.2 Research Question and Objective

According to the aim of the company and research criterion, we can formulate the research question:

To what extent can the amount of model data be reduced through active learning without decreasing the classification accuracy?

Consequently, this research question has its own objectives from different points of views. From the perspective of the degree project, firstly, we will assess to which degree a conventional algorithm can solve the research question. Secondly, we will generate an algorithm, which combines RNNs and pool-based active learning for short-text classification. As a result, we will test whether the novel algorithm can decrease the amount of labeled data samples while keeping good classification accuracy of deep learning algorithm for short-text classification or not. Thus, the success of combining active learning and deep learning will reduce the difficulty of application of RNNs.

From the principal side, we hope that the new algorithm is able to reduce the dependency of short-text classification on labeled data. If this algorithm is efficient, it will overcome the bottleneck of labeling data manually, and consequently it will improve the efficiency of the company's intelligent custom service system for new clients in the future.

## 1.3 Related Work

The key point behind this project is how to use the idea of active learning in the RNNs structure, i.e., achieving greater accuracy with fewer labeled training data for the RNNs classifier. We will briefly introduce RNNs and Gated Recurrent Units (GRUs) in chapter 2, section 2.1. In this section, we focus on related work of active learning.

The key process of acitve learning is selecting new training samples, i.e., it is a sampling process. It is called "Query Strategy Frameworks" in a literature survey [8] for selecting new training samples. These samples are more informative or useful than other samples according to a certain criterion. Previous work in active learning for selecting informative samples or constructing samples selecting criteria has concentrated on at least three approaches: (1) uncertainty based methods, such as least confident [6], margin sampling [10], entropy [11], query by committee [12] [13] and version space reduction [14] [15]; (2) expected model change methods, such as "expected gradient length" (EGL) [16]; (3) underlying distribution based methods, such as clustering-based active learning [17] and hierarchical sampling [18].

There are also some mixed methods that exploit criteria in selecting new training samples for active learning. For example, the density-weighting method [9] applies both uncertainty and density, i.e., the distribution of the unlabeled samples. We will discuss the details about this method in chapter 2 since it is suitable for our application situation. In this project, we propose a novel algorithm both employing uncertainty and distribution for selecting informative samples. The novel algorithm is inspired by density-weighting method. A related method, sampling by uncertainty and density (SUD) [19], where density is calculated through K-Nearest-Neighbor (KNN), was proposed in the same year (2008). Moreover, the article by Zhou et al. [20] proposed Information ADN (active deep network) for sentiment classification, which applies the distance between the sample and the separation line as the measurement of uncertainty and uses the averaged Euclidean distance between the sample and other samples in the same class based on features space (extracted through DBN) as informativeness. In fact, the main idea of sampling in these four methods, including the method proposed in this project, is the same, and the differences pertains to estimating the uncertainty and density or representativeness. In this project, we use item representativeness describing underlying distribution of samples.

Indeed, active learning has been used in many applications in text classification, such as Expectation-Maximization (EM) with active learning (uses a modified QBC) for text classification [21], SVM with active learning (uses a version space reduction) for text classification [15] and so forth. But combining active learning with RNNs classifier for short-text (Chinese) classification is rare.

# Chapter 2

# Methods

In this project, RNNs with GRUs are adopted as the basic classifier for the multi-class short-text classification problem. We exploit pool-based active learning technology under the framework of RNNs in order to study whether active learning could reduce the size of training data set without significantly influencing the accuracy of classification result. We give an overview of the related theory applied in thesis work and propose our novel algorithm based on a deep learning classification architecture.

This chapter is organized as follows: In section 2.1, we describe the application scenarios of active learning, and analyse which scenario is suitable for this project. In section 2.2, we discuss the criteria for sampling in active learning process. Both uncertainty and representativeness, as critera, are discussed in subsection of section 2.2. Furthermore, we innovate a novel algorithm to evaluate or calculate the "representativeness" in 2.2.3. In section 2.3, we introduce the basic idea of RNNs with GRUs, which is the machine learning model used in this project, used in the process of active learning. RNNs provide useful information for active learning, since it acts as a classifier and a feature extracter in this project. Finally, in section 2.4, we describe the idea of cross validation and how to use cross validation in this project.

## 2.1 Pool-Based Active Learning

As mentioned in the backgroud, there are three application scenarios. The application scenario of this project is pool-based, since large collections of unlabeled data are gathered at once [8]. We denoted active learning applied in pool-based scenario "pool-based active learning". Essentially, active learning is an iterative process of selecting the most informative samples to get their labels from unlabeled samples [27]. Hence, application scenarios are irrelevant to iterative sampling process, i.e., the iterative sampling process is the same among the three application scenarios in the background.

Typically, the pool-based active learning can be described as the following steps [28] and it is illustrated in Figure 2.1 [8]:

(1) Start with a pool of unlabeled data.

(2) Pick a certain number of points at random (or through unsupervised learning method, such as clustering [19]) and get their labels.

Repeat (3) and (4),

(3) Learn a classifier based on the labeled data seen so far.

(4) Query the unlabeled point that is closest to the boundary (or most uncertain, or most likely to change the model, ...)

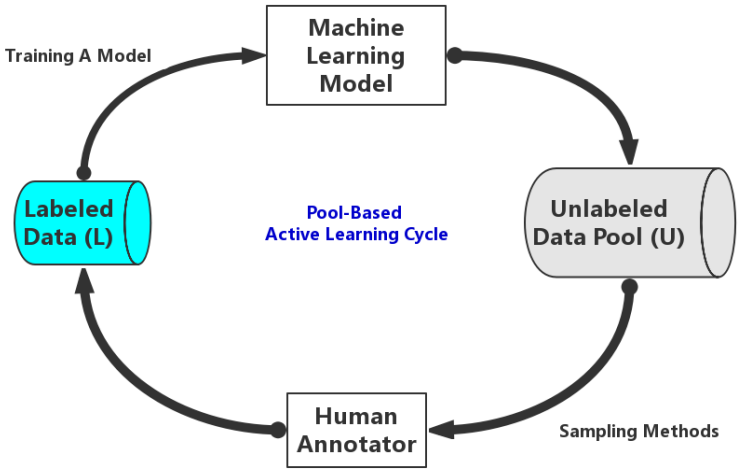


Figure 2.1: The pool-based active learning cycle

As showed in above figure, the real active learning process needs the participation of oracles or human annotators. Thus, the period of active learning is long. In order to verify the validity of active learning in an efficient way, the company labeled all the collected data in advance. But we employ these data as unlabeled data to imitate the process of active learning. So, when we need the true labels (human annotated) of selected samples in above $step\ (4)$, we can get their labels immediately and do not have to stop the program on GPU for asking the labels of selected samples from human annotators. Besides, the machine learning model shown in Figure 2.1 is RNNs with GRUs in this project. We will briefly describe it in section 2.3. The key point of this project is selecting an efficient sampling method in active learning under the RNNs architecture.

We are eager to achieve the goal that the classifier trained by part of the whole data set has similar classification result as the classifier trained by the whole data. Therefore, the $step\ (4)$ is the most important step in active learning process. It decides which are the most useful samples or most informative samples to be labeled. If the sampling strategy is perfect, the training data set will be as small as possible. Taking support vector machine in classification problem for example, if we select support vectors as training data set at once, it performs as well as using the whole data set. So, the kernel procedure of active learning is how to select informative samples or how to sampling. The criteria for sampling are motivated and are discussed in following section.

## 2.2 Criteria for Sampling

In this section, criteria for selecting training samples, i.e., sampling methods, are discussed, which consist in the conventional algorithms and our novel algorithm. As mentioned in last section, the kernel procedure of active learning is selecting samples to get their labels. In statistics, when choosing a point to learn a function, it is important to choose the "most informative point" [29]. It is the same in machine learning when we

need to build up a learner, since the process of building machine learning model is effectively same as a process of fitting a function from inputs to outputs. Consequently, it is important to choose the most informative sample for learning a model. We select these informative samples based on their informativeness. The following algorithms are criteria for calculating informativeness of samples. We do not list all the criteria for sampling listed in section `related work`. Here, we only introduce the methods needed in this project.

### 2.2.1 Random Sampling

Random sampling is the most common sampling method in scientific researches. Because, random sampling can indistinguishably select the samples through the whole data set. It will represent the distribution of original data by few data samples. To a certain extent, random sampling is useful to describe the original data in an universal way. As shown in Figure 2.1, we should select samples to be labeled. The sampling method can be random sampling. In random sampling, each sample has the same probability as others to be selected.

### 2.2.2 Uncertainty Sampling

Random sampling considers that there is no differece between samples. However, some samples have more information than others. Thus, selecting these informative samples to learn a classifier or to fit a function is more significant than selecting samples randomly.

As mentioned in section `related work`, there are several methods evaluating the informativeness of unlabeled samples. The most intuitive and common one for this project is uncertainty sampling [6], since the multi-classifier constructed by RNNs is a probabilistic classifier, i.e., it outputs a vector of probabilities. Each element in this vector is the probability that the sample belongs to one certain class. Uncertainty is computed from these probabilities. More clearly, if there are $C$ classes, the output of RNNs model will be a vector $P = [p_1, p_2, ..., p_C]$, $\sum_{i=1}^{C} p_i = 1$, and $p_i$ means the probability that the sample belongs to class $i, i \in [1, 2, .., C]$.

There are at least three uncertainty sampling methods, consisting of least confident [9], margin sampling [10], and entropy [11]. Least confident strategy only considers information about the most probable label, while margin sampling considers information about the first two most likely labels. This project has $435$ classes. In order to avoid omitting information about most information reflected in label distribution, we choose entropy as the measure of uncertainty. It is generally used as a measure of uncertainty or impurity or informativeness in machine learning [6] [8]. The following formula is used for selecting informative samples through entropy.

$$x^* = \underset{x}{argmax} - \sum_{i=1}^{C} P_\theta(\hat{y_i}|x) log P_\theta(\hat{y_i}|x)$$

where $x^*$ is the most informative instance; $P_\theta(\hat{y_i}|x)$ is the probability that the sample $x$ belongs to the $i$th class $y_i$ under the model $\theta$, i.e., $\hat{y_i}$ is estimated class through current model; $C$ is the number of classes.

For the sake of reducing computational overhead, in this project, we calculate entropy through the first $10$ most probable labels. Empirically, the first 10 most probable labels

occupies $95\%$ information of total $435$ labels. The test accuracy which is based on first $10$ most probable labels is in range $[95\%, 100\%]$. In other words, when we predict the label of a sample according to the trained classifier, the probability that the real label is in the first $10$ most probable labels generated from the classifier, is more than $95\%$. Consequently, the computation of entropy is adjusted in this project and is reflected in the following formula.

$$x^* = \underset{x}{argmax} - \sum_{i=1}^{10} P_\theta(\hat{y}_i|x) log P_\theta(\hat{y}_i|x)$$

where, the probabilities of the first $10$ predicted classes under model $\theta$ are is in descending order, i.e., $P_\theta(\hat{y_1}|x) \geq P_\theta(\hat{y_2}|x) \geq ... \geq P_\theta(\hat{y_{10}}|x)$, and $\hat{y}_i$ means one estimated class of $435$ classes.

### 2.2.3 Density-Weighted Method

Only using uncertainty to selecting informative samples has weaknesses. For example, selecting an outlier, i.e., a sample does not inhabit dense regions of the input space (or feature space), and putting it into training data set will not be of benefit to train model even through this sample is uncertain. The problem can be illustrated in Figure 2.2, where black samples means they belongs to class black and have black labels, as with yellow samples, while white samples mean unlabeled samples. In this circumstance, we prefer to select the sample $A$ as the new training data, which is "representative", i.e., inhabit dense regions of the input space, and uncertain, i.e., close to the decision boundary. Because sample $A$ inhabit a dense region of the input sapce (or feature space), while sample $B$ has less neighbor samples, and consequently misclassifying sample $A$ will influence more samples than misclassifying sample $B$. In other words, we have more interests on query the label of sample $A$ in order to avoid misclassifying more samples.

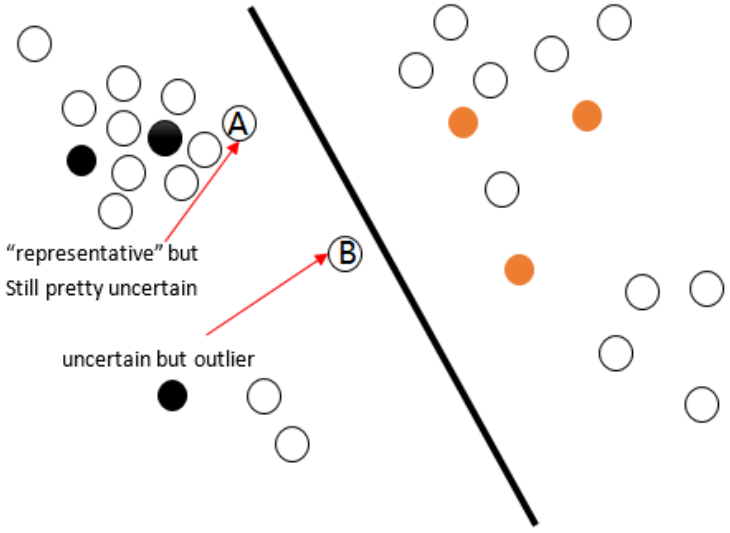


Figure 2.2: Problem of Uncertainty Sampling: Outliers

So, a general density-weighted method with the information density framework was proposed by Settles and Craven [9]. It states that the most informative samples should

not only be those which are uncertain, but also those which are representative of the underlying distribution [8]. Therefore, the most informative samples are calculated from the following formula.

$$x^* = \underset{x}{argmax}\ \Phi(x)\frac{1}{U}\sum_{u=1}^{U} sim(x, x^{(u)})$$

where, $\Phi(x)$ means the uncertainty of candidate sample $x$, which is described in last section; $U$ represents the number of samples in unlabeled pool $U$; $sim(x, x^{(u)})$ measures the similarity between candidate sample $x$ and arbitrary sample $x^{(u)}$ in unlabeled pool. A cosine similarity function was used as a similarity function in [9].

At present, the density-weighted method is an ideal algorithm for this project. On the one hand, this project exploits a probability multi-classifier for $435$ classes classification problem. On the other hand, abundant training data set has a certain number of outliers. The density-weighting method considers both uncertainty and representativeness. It is a synthetical algorithm, but should be adjusted in this project. As mentioned in last section, entropy is calculated from top $10$ classes of $435$ based on the predication. The representativeness of a sample can not be directly computed through formula: $\frac{1}{U}\sum_{u=1}^{U} sim(x, x^{(u)})$. Because, the size of unlabeled data pool is huge. If we compute the similarity of each two samples and then get their average, the computational overhead will be enormous.

Inspired by above density-weighted method, a method to estimate representativeness of one sample for RNNs architectures is proposed in this paper. It can reduce computational overhead in industry application. Besides, it is simple and general for all kinds of deep learning problems combined with active learning. It will be intruced in following section.

### 2.2.4 Weighted Uncertainty Sampling

In the previous section, we defined the representativeness based on similarity and the uncertainty measurement by least confident, margin, and entropy. In this section, the novel discriminate function is constructed through representativeness and uncertainty, which is applied to selcet the most informative samples. The process of computing representativeness in this novel algorithm requries RNNs, which will be introduced briefly in next section.

The process of computing representativeness can be summaried in following recurrent steps:

(1) Train a RNNs classifier on the basis of labeled data.

(2) Apply the classifier on unlabeled data, generate the vector of probabilities for prediction and the hidden vetor from hidden layer of RNNs as features of each sample.

(3) For uncertainty of one sample, we compute the entropy of ranked probabilities of top 10 classes based on the classifier.

(4) For representativeness of one sample, there is a process: First, we use the estimatied label from the classifier as the class of the sample. Then for each class, we compute the center in feature space generated from RNNs hidden layer. Finally, we compute

the Cosine similarity between the sample and its class center and normalized the result from range $[-1, 1]$ to range $[0, 1]$, since we need the same scale to measure representativeness as uncertainty.

(5) Multiply the uncertainty and representativeness and regard the product as the judgement of selecting samples. Exploit descending order on these products and select top certain number of samples.

(6) Label these selected samples and put them in training data set.

Here, we named the product of uncertainty and representativeness of one sample as "weighted-uncertainty" (WU). So, the sampling method of active learning is named as "weighted-uncertainty sampling". It can be formulated in following formulas.

$$\begin{cases} Ent(x) = -\sum_{i=1}^{10} P_\theta(\hat{y}_i|x) log P_\theta(\hat{y}_i|x) \\ C_i = \frac{1}{N_i} \sum_{j=1}^{N_i} h_j^{(i)}, i \in [1, 2, .., C] \\ Sim(x) = Cosine\ Similarity(h^{(i)}, C_i) = \frac{h^{(i)} \cdot C_i}{\|h^{(i)}\| \|C_i\|} \\ WU(x) = Ent(x) \cdot Sim(x) \end{cases}$$

where,

- $Ent(x)$ means the uncertainty of candidate sample $x$, which is described in section 2.3.1;
- $x^{(i)}$ illustrates candidate sample $x$ belongs to class $i$ on basis of trained classifier, and $x_j^{(i)}, j \in [1, 2, ..., N_i]$ means all samples belong to class $i$ according to trained classifier; obviously, $N_i$ is the number of samples in class $i$, and $N_i$ might be different from each other, i.e., different class has different number of samples; $h_j^{(i)}$ is the hidden vector in feature space of one input sample $x_j^{(i)}$; $C_i$ means the center of class $i$ in feature space, and $C$ represents the number of classes;
- $h^{(i)}$ is the hidden vector corresponding to input sample $x^{(i)}$, and $Sim(x)$ means the simlilarity between $h^{(i)}$, the hidden vector of $x$ when $x$ is classified to class $i$, and the class center $C_i$. $Sim(x)$ can be regarded as the weight of the sample $x$.
- $WU(x)$ is the weighted uncertainty of sample $x$, product of uncertainty and representativeness.

So, the candidate samples are generated from following formula:

$$x^* = \underset{x}{argmax}\, WU(x)$$

The whole process of proposed active learning algorithm can be illustrated through Figure 2.3.

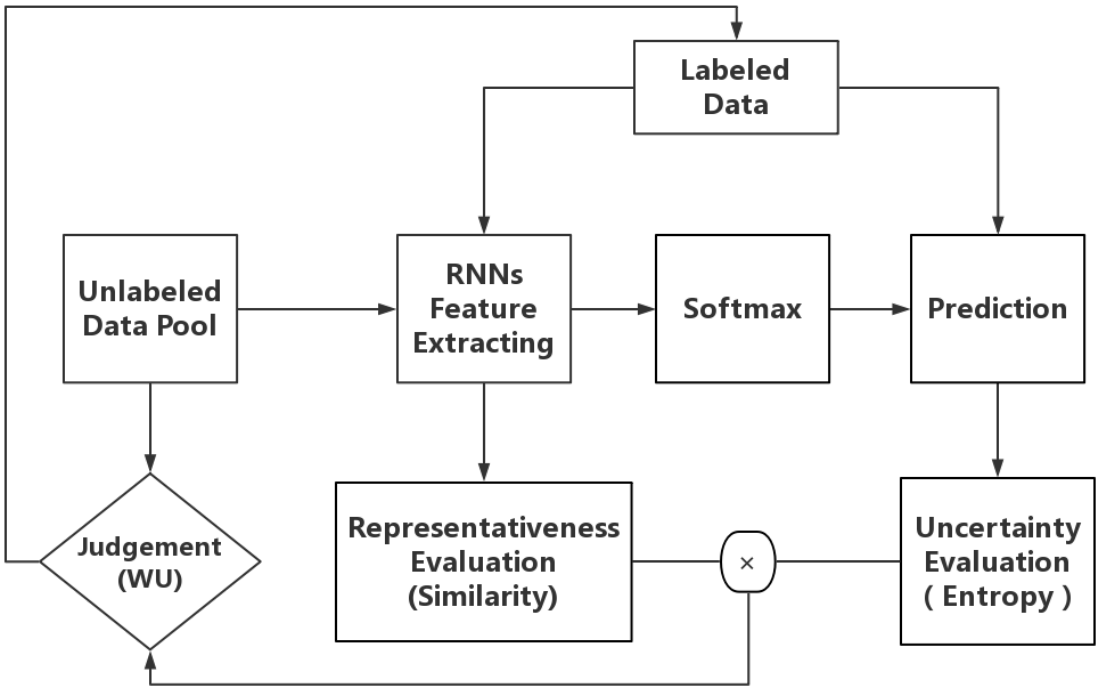


Figure 2.3: Proposed active learning algorithm

## 2.3 RNNs with GRUs

In this project, we use Recurrent Neural Networks (RNNs) [4] [22] combined with Softmax Regression [23] to generate a multi-classifier for short-text. RNNs are responsible for feature extracting of short-text, while Softmax is used for classification, which are illustrated in Figure 2.3. The output of the classifier is a vector of probabilities and the length of the vector is the number of total classes. Each element in this vector is a probability, which represents the probability that a sample belongs to the corresponding class. So, the final class label generated by the classifier of the input sample is the class label with largest probability in the vector. In the following, we describe the typical RNNs.

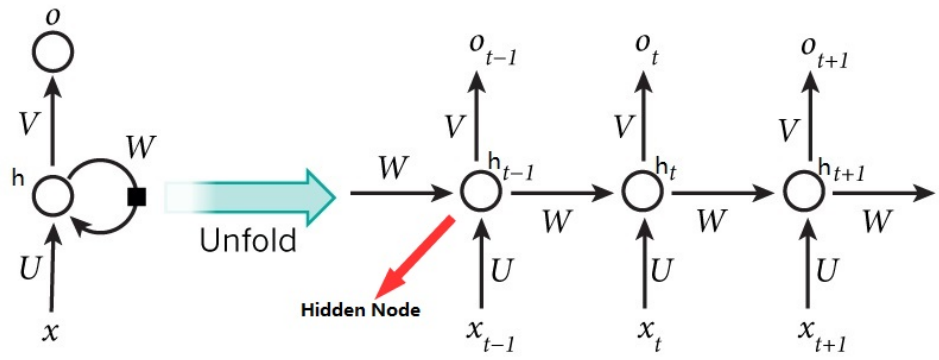


Figure 2.4: A recurrent neural network and its unfolding structure based on time

where,

- $x_t$ is the input at time step $t, t \in [1, 2, ..., T]$, $T$ is the number of total time steps, i.e., the length short-text. The network cannot cope with text directly. In this project, short-text means the text has no more than $35$ words according to the company, and consequently $T = 35$. Firstly, we split the short-text into $35$ words. If one short-text has more than $35$ words, we cut the excess off. On the contrary, if one shor-text has less than $35$ words, we will make up the short-text to $35$ through the special words. Then we use $word2vec$ technology [24] which codes a word as a $300$ dimension vector, and the special word corresponds to zero vector. In other words, $x_t$ is a vector representing a single word or phrase. At present, the company has its own $word2vec$ table, which has $65,447$ words, including phrases, and their corresponding vectors. So, the actual input $x$ of RNNs is a matrix with size $35 \times 300$. If we find some new words or phrase, we will update the $word2vec$ table.

- $h_t$ is the hidden state at time step $t$. Unlike a traditional deep neural network, $h_t$ relies on both the previous hidden state and the input at current time step: $h_t = f(Ux_t + Wh_{t-1})$. Empirically, the function $f$ usually is $tanh$ or $ReLU$, which is a nonlinear function.

- $o_t$ is the output at step $t$. In this project, as we want to classify the short-text, we can calculate the output: $o_t = softmax(Vh_t)$. It is a vector of probabilities and each element in the vector signifies how probable the input simple belongs to corresponding class.

- $U, V, W$ are shared across all time steps as the same parameters in RNNs.

According to the architecture of RNNs, we can say that using RNNs is suitable for short-text's feature extraction and classification. Because a short-text can be regarded as a continues time sequence where there exists relationships between words in time order.

More specifically, for the purpose of overcoming vanishing gradient problem in standard RNNs [4] [25], the standard hidden nodes of RNNs marked in Figure 2.4 are replaced by Gated Recurrent Units (GRUs) [5] [26] shown in Figure 2.5 [26].

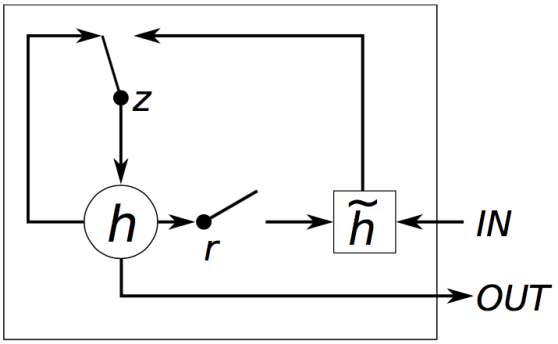


Figure 2.5: GRU Gating, which is the replacement of the standard hidden nodes in RNNs

We should clearly know that a GRU layer just exploits another point of view to compute a hidden state $h_t$. Previously, we computed the hidden state as $h_t = \tanh(Ux_t +$

$Ws_{t-1}$). The inputs to this unit were $x_t$, the current input at step $t$, and $s_{t-1}$, the previous hidden state. The output was a new hidden state $s_t$. A GRU unit does the exact same thing in a different way. Figure 2.6 shows the computational process of a GRU.

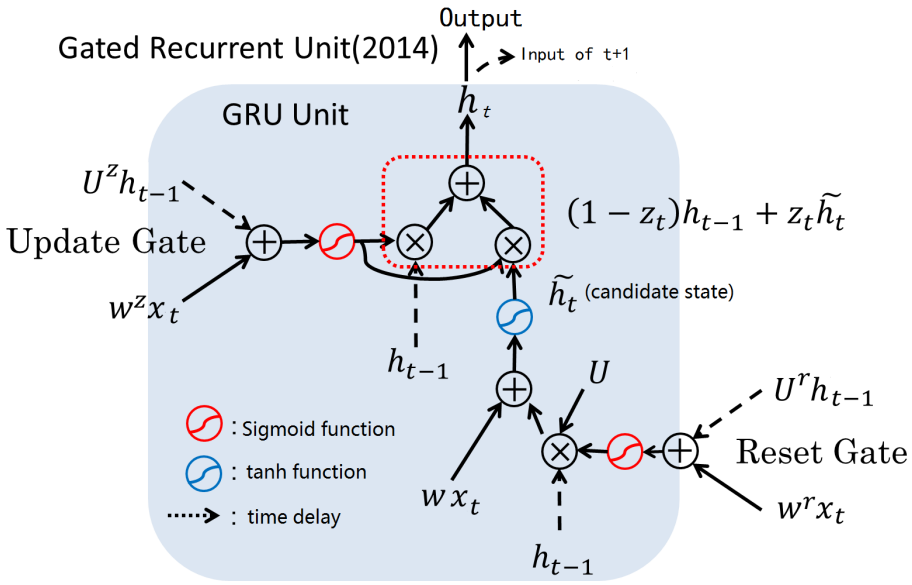


Figure 2.6: GRU computational process. Figure Source: "https://qiita.com/" (Qiita is a technical knowledge sharing and collaboration platform for programmers.)

So, the following equations illustrate the process of computing hidden state $h_t$ through a GRU.

$$\begin{cases} z_t = \sigma(W^z x_t + U^z h_{t-1}) \\ r_t = \sigma(W^r x_t + U^r h_{t-1}) \\ \widetilde{h_t} = \tanh(W x_t + U(h_{t-1}\, r_t)) \\ h_t = (1 - z_t)h_{t-1} + z_t \widetilde{h_t} \end{cases}$$

where, $r$ is the reset gate and $z$ is the update gate.

A detailed description of how to train RNNs with GRUs is in [4] and Chung et al. [26]. The training method is backpropagation through time (BPTT).

## 2.4 Cross Validation

Cross Validation is a model validation method for evaluating how the results of a statisticla analysis (or machine leanring) will generalize to an independent data set. In this project, we exploit the k-fold cross validation which is the most popular one among cross validation. The process of k-fold cross validation can be described as following steps:

(1) First, separate at random the total data set into $k$ subset with similar size and without any intersection.

(2) Then use $k-1$ subsets as training data set to train a model and use the last subset to test the model.

(3) Repeat step (2) $k$ times and get the average test error (or accuracy).

(4) Finally, the average test error (or accuracy) is the criterion of the generalization of this model.

In this project, we use 5-fold cross validation. Figure 2.7 illustrates the process of 5-fold cross validation. There are at least two reasons for selecting $5$ as the value of $k$ in $k$-fold cross validation. On the one hand, the total data set is big. If $k$ is large, the computational overhead will be huge, which indicates wasting time and money. On the other hand, the experiential value of $k$ is $5$ or $10$. So, $k = 5$ is an experiential value in this project, which comes from the consideration of computational overhead and the efficiency of cross validation.

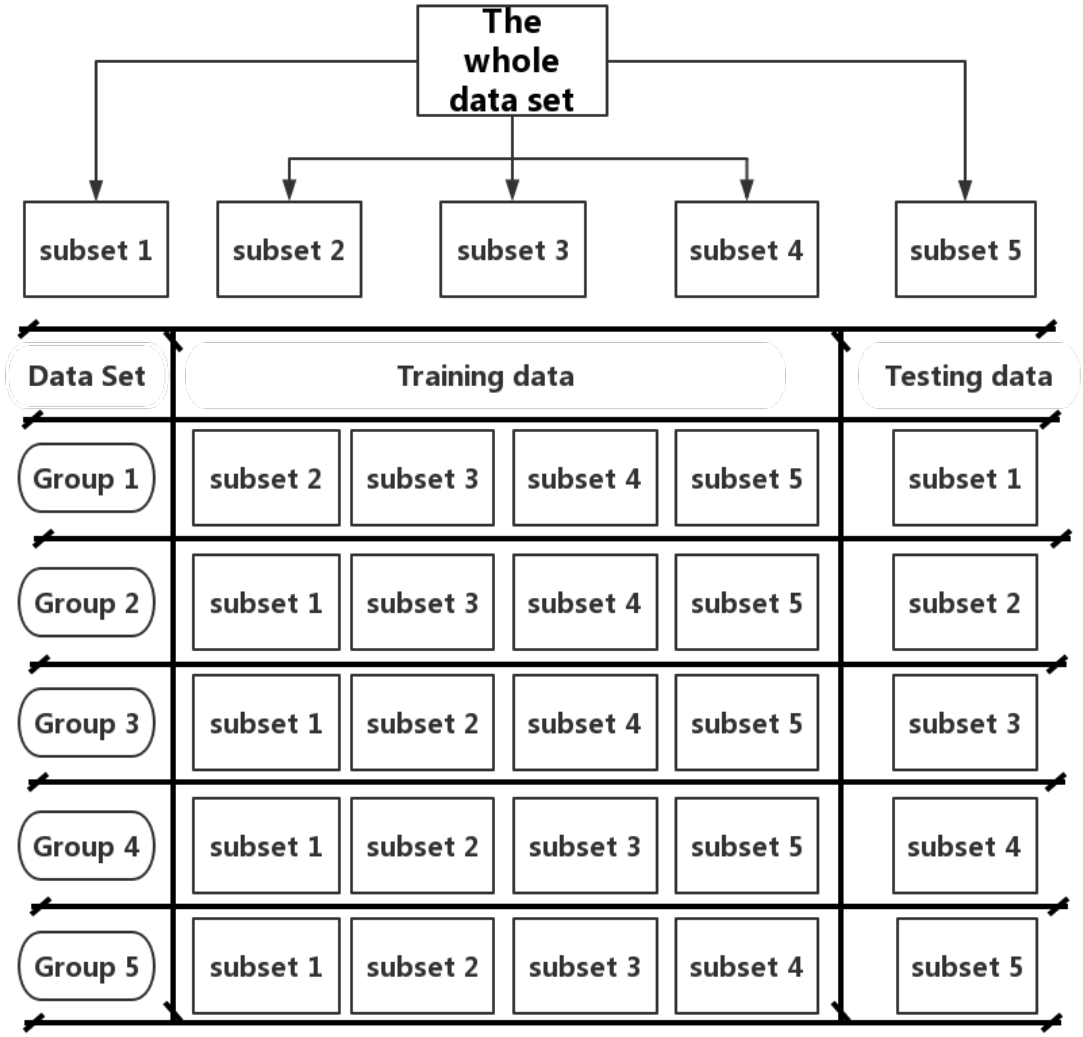


Figure 2.7: 5-fold cross validation used in this project

In following chapters, we use $Group\ 1$ to $Group\ 5$ to name the data set, i.e., $Group\ 1$ consists of training data set 1, including subset 2 to subset 5, and testing data set 1, including subset 1. And, $Group\ 2$ consists of training data set 2, including subset 1, subset 3, subset 4 and subset 5, and testing data set 2, includes subset2, et cetera. The name of

data sets are reflected in Figure 2.7.

In this project, we have $73,747$ short-text samples. These samples are belongs to $435$ classes. These classes are the topics of short-text samples. For example, "I want to book a car at 10:00" belongs to topic "booking". On average, each class has around $170$ samples. According to 5-fold cross validation, each subset has around $14,750$ short-text samples. So, the training data set has around $59,000$ short-text samples while the testing data set has about $14,750$ samples. The distribution of training data and testing data are similar since we seperate the original data set randomly.

# Chapter 3

# Experiments and Results

In this chapter, we introduce the designed experiments based on four different classification models, original RNNs classifier, active RNNs classifier with random sampling, active RNNs classifier with uncertainty sampling and active RNNs classifier with weighted uncertainty sampling. Original RNNs classifier is built up by the company, but they did not do the 5-fold cross validation. Besides, the original classifier gives an architecture for features extracting and class prediction for active learning. We present corresponding results based on above models. All of these models are verified through 5-fold cross-validation. The original RNNs classifier is a baseline to test whether active learning could reduce the amount of training data set without significantly influencing the classification accuracy. Meanwhile, we can compare the performances of different sampling stategies, i.e. random sampling, uncertainty sampling and weighted uncertainty sampling, applied in active learning based on RNNs classifier architecture.

There are at least two criteria or two points of view to evaluate whether active learning or sampling method makes sense: (1) Set an acceptable value based on original classification accuracy as the lower limit goal, such as $80.00\%$, and research how much data active learning can reduce meanwhile satisfying the classification accuracy; (2) Set a goal about how much data we want to reduce, such as $50\%$ of original data, and check whether the classification accuracy is acceptable based on part of original training data selected through active learning strategy. In this project, we choose criterion one in order to evaluate the efficiency of active learning. Furthermore, we define that if we have exploited $80\%$ of original training data and wouldn't achieve an acceptable classification accuracy described in criterion one, we consider active learning invalid for our project.

According to above sampling criteria, we organize this chapter as follows: In section 3.1, we test the original RNNs classifier on total training data set and get a baseline of classification accuracy. Besides, we used some small tricks to reduce the computational overhead, such as "early stop". In section 3.2, we use active RNNs classifier with random sampling to set a baseline of active learning. In other word, the sampling method introduced in section 3.3 and 3.4 must be better than random sampling. In section 3.3, active RNNs model with uncertainty sampling is applied to represent as traditional active learning strategy. Specifically, the original RNNs model acts as a classifier only, and the uncertainty is calculated just on basis of classification result. So, the RNNs classifier could be replaced by any other classifier for short-text classification, such as SVM. In section 3.4, we use RNNs classifier not only as a classifier to compute the uncertainty, but also as a features extracter to compute the representativeness of arbitrary sample. Fur-

thermore, we use the product of uncertainty and representativeness as criterion for selecting sample in active learning. The key point is that we use RNNs to extact the features of samples. We use this active learning strategy in RNNs architecture to research whether we can reduce the amount of training data.

## 3.1 RNNs model

Before showing the result of original RNNs classifier, we should clearly state that:

(1) According to the process of 5-fold cross validation, we get five accuracy lines on five test sets with training epoch as x-axis and test accuracy as y-axis in Figure 3.1 and five columns in Table 3.1.

(2) We set an "early stop" mechanism to save the computational overhead. If the classification accuracy won't increase in further 10 training epochs, we stop training the model. So, you can see that some lines in Figure 3.1 are shorter than others and some columns in Table 3.1 has fewer rows than others since some training processes are early stopped. The details aboute "early stop" are shown in Figure 3.2.

(3) Empirically, completely training a RNNs model needs around 28 steps based on the data set of the company. We set 39 steps for training a RNNs classifier considering both completely training and "early stop" mechanism.

So, both Table 3.1 and Figure 3.1 describe the results of 5-fold cross validation of original RNNs classifier in details. Furthermore, Table 3.2 summaries the 5-fold cross valdiation of RNNs model and Figure 3.2 reflects the details of "early stop" in each data set.

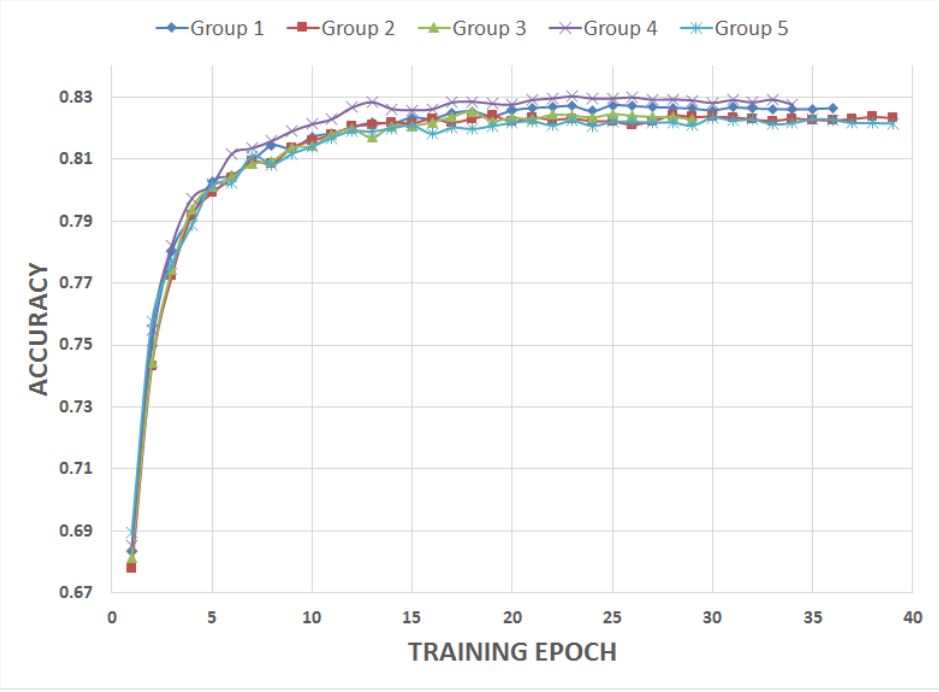


Figure 3.1: 5-fold cross validation of RNNs Classifier

**Notes:** 1. The accuracies reported in above figure are test accuracies, i.e., the RNNs model is trained through training data and tested through testing data. 2. The goal of cross validation is finding the best model, which can get the best test accuracy. That means

| Epoch | Test Accuracy (Group 1) | Test Accuracy (Group 2) | Test Accuracy (Group 3) | Test Accuracy (Group 4) | Test Accuracy (Group 5) |
|---|---|---|---|---|---|
| 1 | 0.6834 | 0.6780 | 0.6814 | 0.6851 | 0.6896 |
| 2 | 0.7499 | 0.7431 | 0.7443 | 0.7547 | 0.7576 |
| 3 | 0.7802 | 0.7724 | 0.7744 | 0.7822 | 0.7762 |
| 4 | 0.7917 | 0.7919 | 0.7940 | 0.7973 | 0.7886 |
| 5 | 0.8028 | 0.7991 | 0.8012 | 0.8018 | 0.8016 |
| 6 | 0.8048 | 0.8038 | 0.8042 | 0.8117 | 0.8023 |
| 7 | 0.8094 | 0.8092 | 0.8084 | 0.8135 | 0.8111 |
| 8 | 0.8145 | 0.8085 | 0.8094 | 0.8159 | 0.8083 |
| 9 | 0.8134 | 0.8138 | 0.8138 | 0.8190 | 0.8117 |
| 10 | 0.8172 | 0.8160 | 0.8142 | 0.8212 | 0.8140 |
| 11 | 0.8184 | 0.8178 | 0.8180 | 0.8229 | 0.8169 |
| 12 | 0.8204 | 0.8204 | 0.8197 | 0.8267 | 0.8191 |
| 13 | 0.8218 | 0.8212 | 0.8170 | 0.8284 | 0.8189 |
| 14 | 0.8216 | 0.8218 | 0.8208 | 0.8262 | 0.8200 |
| 15 | 0.8234 | 0.8216 | 0.8208 | 0.8258 | 0.8208 |
| 16 | 0.8231 | 0.8230 | 0.8219 | 0.8261 | 0.8182 |
| 17 | 0.8249 | 0.8218 | 0.8237 | 0.8283 | 0.8201 |
| 18 | 0.8254 | 0.8231 | 0.8254 | 0.8286 | 0.8197 |
| 19 | 0.8239 | 0.8241 | 0.8223 | 0.8279 | 0.8207 |
| 20 | 0.8258 | 0.8220 | 0.8234 | 0.8276 | 0.8217 |
| 21 | 0.8265 | 0.8234 | 0.8229 | 0.8291 | 0.8223 |
| 22 | 0.8269 | 0.8227 | 0.8244 | 0.8296 | 0.8209 |
| 23 | 0.8271 | 0.8231 | 0.8241 | 0.8303 | 0.8224 |
| 24 | 0.8255 | 0.8223 | 0.8235 | 0.8296 | 0.8207 |
| 25 | 0.8274 | 0.8222 | 0.8246 | 0.8296 | 0.8220 |
| 26 | 0.8271 | 0.8212 | 0.8239 | 0.8299 | 0.8221 |
| 27 | 0.8268 | 0.8223 | 0.8236 | 0.8292 | 0.8217 |
| 28 | 0.8266 | 0.8241 | 0.8236 | 0.8292 | 0.8218 |
| 29 | 0.8263 | 0.8236 | 0.8228 | 0.8290 | 0.8208 |
| 30 | 0.8258 | 0.8237 | - | 0.8282 | 0.8234 |
| 31 | 0.8267 | 0.8235 | - | 0.8290 | 0.8225 |
| 32 | 0.8263 | 0.8229 | - | 0.8284 | 0.8228 |
| 33 | 0.8262 | 0.8223 | - | 0.8292 | 0.8213 |
| 34 | 0.8261 | 0.8231 | - | 0.8276 | 0.8216 |
| 35 | 0.8262 | 0.8227 | - | - | 0.8229 |
| 36 | 0.8265 | 0.8225 | - | - | 0.8227 |
| 37 | - | 0.8229 | - | - | 0.8216 |
| 38 | - | 0.8236 | - | - | 0.8216 |
| 39 | - | 0.8232 | - | - | 0.8215 |
| Early Stop? | Yes | No | Yes | Yes | No |

Table 3.1: 5-fold cross validation of Original RNNs Classifier

the best model has good generalization in application. As a result, we can exploit the best model as the final model in real world. The average classification accuracy just reflects the performance of the models in statistic on current data set. So, we report both

the separate accuracy for each Group and the average accuracy.

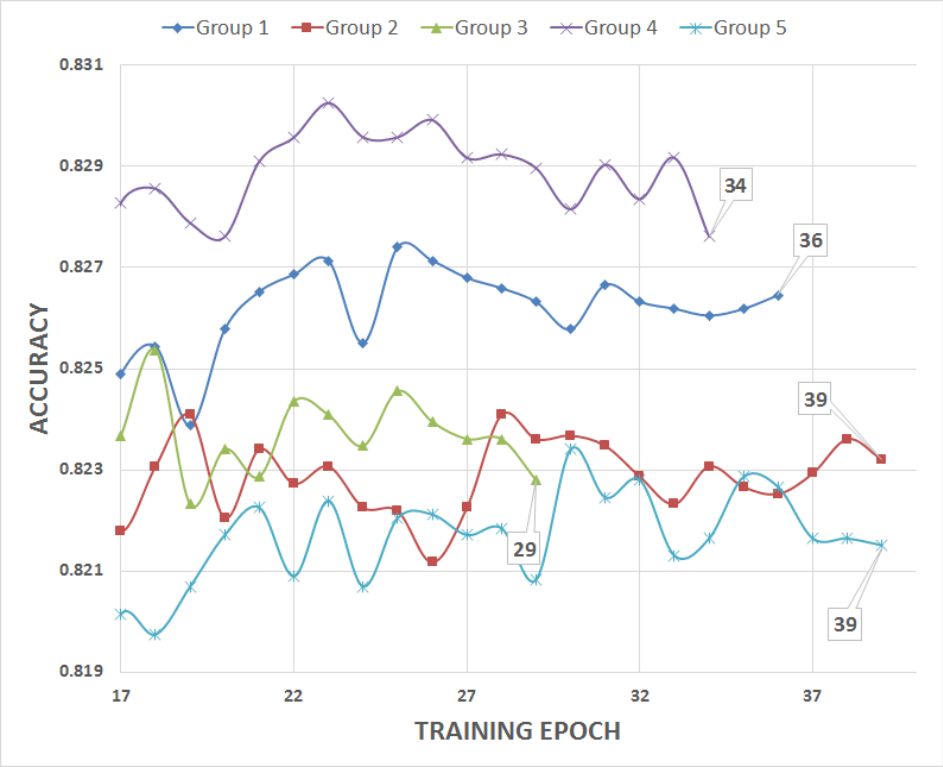


Figure 3.2: Details of 5-fold cross validation of RNNs Classifier from Epoch 17

**Notes:** The number marked in above figure reflect the total training epochs of each data set.

| Data Set | Best Test Accuracy | Best Training Epoch (Getting the Best Test Accuracy) |
|---|---|---|
| Training Set 1 and Testing Set 1 | 82.74% | 25 |
| Training Set 2 and Testing Set 2 | 82.41% | 28 |
| Training Set 3 and Testing Set 3 | 82.54% | 18 |
| Training Set 4 and Testing Set 4 | 83.03% | 23 |
| Training Set 5 and Testing Set 5 | 82.34% | 30 |
| Average test accuracy | 82.61% | 25 |

Table 3.2: Summary 5-fold cross validation of RNNs Classifier

In conclusion, the average classification accuracy of RNNs classifier is $82.61\%$, which is computed through best test accuracies based on $5$ test sets. Because, we consider the classifier is completely trained when it generates the best test accuracies. Furthermore, we can define that the acceptable accuracy range is $[81.61\%, 100\%]$ for active learning classifier. The acceptable difference is $1\%$. In other word, if the active RNNs classifier acheives test accuracy $81.61\%$ with less data (less than $80\%$ of original data), we regard the active learning method is feasible. This "acceptable criterion" (difference is $1\%$) is defined on the basis of questionnaire survey of top artificial intelligence companies and their customer in China. It is a "criterion" in application level without any unified stan-

dards. The other thing should be studied is how valid active learning method is, i.e., how much data can active learning method reduce under the accuracy limitation. It will be discussed in following sections.

## 3.2 Active RNNs model with Random Sampling

Before showing the results, including table and figure, of active RNNs model with random sampling, we should clearly state that:

(1) The first column in Table 3.3 means the percentages of the whole data (as training data). Moreover, we define that the up limitation of trainind data in active learning is $80\%$ of whole data. So, we set the epoch of active learning epoch $30$. Thus, Table 3.2 has $30$ rows except the title line.

(2) The five columns in Table 3.3 from second column to sixth column represent 5-fold cross validation based on five groups data, which are difined in section 2.4 `cross validation`.

(3) The seventh column indicates the average test accuracy of the RNNs classifiers trained through five current data sets seperately. It is regarded as the final test accuracy of active RNNs classifier according to 5-fold cross validation.

(4) The eighth column represents the standard deviation of average test accuracy.

(5) As mentioned in section 2.2, there is a step about selecting samples from unlabeled data pool to label them. Active RNNs model with random sampling means that we select a certain number of samples randomly.

(6) In order to intuitionisticly see the size of training data, in each active learning epoch, we select $5\%$ from left unlabeled pool and we set the horizontal ordinate in percentages, meaning training data used ratio the whole data in Figure 3.3.

Besides, completely training and "early stop" mechanism considered in training active RNNs models, consisting in active RNNs classifier with random sampling, uncertainty sampling and weighted uncertainty sampling. In other words, active learning methods take charge of selecting training data set, while RNNs classifier is trained based on these data sets and it needs completely training and "early stop" mechanism. We will not mention this in following two sections.

Finally, Table 3.3 illustrates the details in the 5-fold cross valdiation of active RNNs model with random sampling. Figure 3.3 summaries the result. The standard deviation also is reflected in the figure.

According to Figure 3.3, the classification accuracy of active RNNs classifier with random sampling does not approach the acceptable range $[81.61\%, 100\%]$ after employing $80\%$ of the whole data set. But it reflects a fact that with the increment of training data, the classification accuracy increases. This is a baseline for active learning with other sampling methods. In other word, active learnig with other sampling methods should perform better than random sampling.

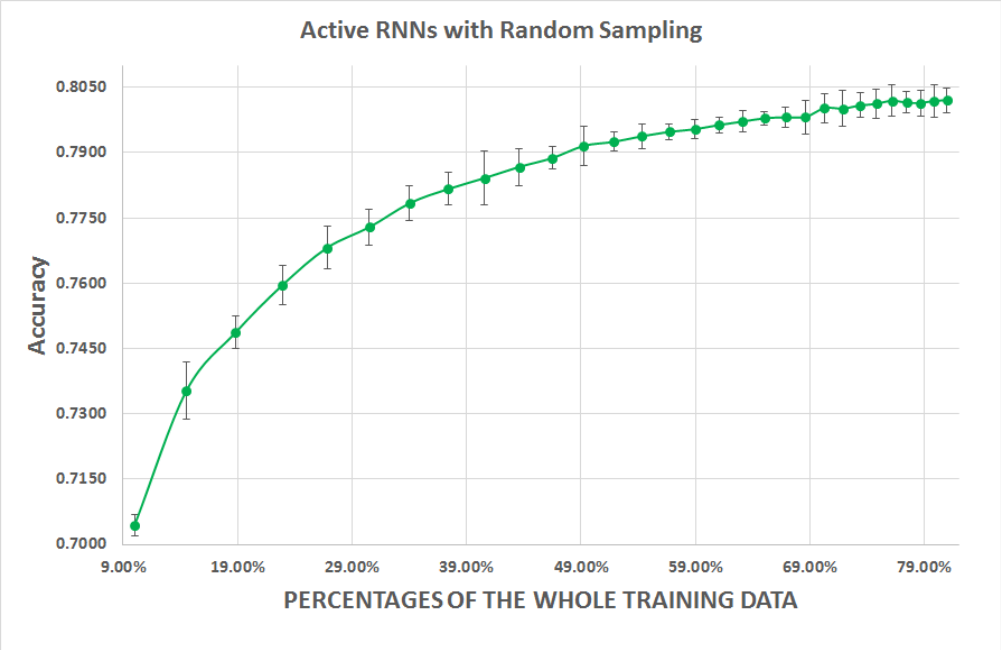


Figure 3.3: Active RNNs Classifier with Random Sampling

| Data Used | Test Accuracy (Group 1) | Test Accuracy (Group 2) | Test Accuracy (Group 3) | Test Accuracy (Group 4) | Test Accuracy (Group 5) | Average Test Accuracy | Standard Deviation |
|---|---|---|---|---|---|---|---|
| 10.00% | 0.7079 | 0.7018 | 0.7049 | 0.7052 | 0.7026 | 0.7045 | 0.0024 |
| 14.52% | 0.7408 | 0.7366 | 0.7342 | 0.7401 | 0.7246 | 0.7353 | 0.0065 |
| 18.83% | 0.7519 | 0.7472 | 0.7468 | 0.7533 | 0.7444 | 0.7487 | 0.0037 |
| 22.92% | 0.7638 | 0.7595 | 0.7581 | 0.7636 | 0.7530 | 0.7596 | 0.0045 |
| 26.81% | 0.7723 | 0.7662 | 0.7651 | 0.7742 | 0.7630 | 0.7682 | 0.0048 |
| 30.52% | 0.7751 | 0.7709 | 0.7695 | 0.7791 | 0.7699 | 0.7729 | 0.0041 |
| 34.04% | 0.7816 | 0.7760 | 0.7773 | 0.7832 | 0.7739 | 0.7784 | 0.0039 |
| 37.40% | 0.7851 | 0.7810 | 0.7788 | 0.7859 | 0.7775 | 0.7817 | 0.0037 |
| 40.58% | 0.7893 | 0.7832 | 0.7822 | 0.7907 | 0.7755 | 0.7842 | 0.0061 |
| 43.62% | 0.7906 | 0.7843 | 0.7852 | 0.7918 | 0.7819 | 0.7868 | 0.0043 |
| 46.50% | 0.7898 | 0.7895 | 0.7878 | 0.7919 | 0.7849 | 0.7888 | 0.0026 |
| 49.25% | 0.7953 | 0.7919 | 0.7885 | 0.7964 | 0.7858 | 0.7916 | 0.0045 |
| 51.86% | 0.7941 | 0.7916 | 0.7906 | 0.7956 | 0.7906 | 0.7925 | 0.0022 |
| 54.34% | 0.7965 | 0.7922 | 0.7923 | 0.7972 | 0.7908 | 0.7938 | 0.0029 |
| 56.70% | 0.7962 | 0.7926 | 0.7950 | 0.7968 | 0.7934 | 0.7948 | 0.0018 |
| 58.95% | 0.7974 | 0.7938 | 0.7934 | 0.7983 | 0.7941 | 0.7954 | 0.0023 |
| 61.08% | 0.7970 | 0.7952 | 0.7954 | 0.7992 | 0.7951 | 0.7964 | 0.0018 |
| 63.11% | 0.7993 | 0.7950 | 0.7962 | 0.8001 | 0.7952 | 0.7972 | 0.0024 |
| 65.04% | 0.7980 | 0.7979 | 0.7965 | 0.8005 | 0.7967 | 0.7979 | 0.0016 |
| 66.88% | 0.7999 | 0.7977 | 0.7957 | 0.8011 | 0.7963 | 0.7981 | 0.0023 |
| 68.63% | 0.8018 | 0.7944 | 0.7963 | 0.8029 | 0.7957 | 0.7982 | 0.0039 |
| 70.29% | 0.8030 | 0.7969 | 0.7982 | 0.8046 | 0.7986 | 0.8003 | 0.0034 |
| 71.87% | 0.8032 | 0.7948 | 0.7982 | 0.8053 | 0.7992 | 0.8001 | 0.0042 |
| 73.38% | 0.8007 | 0.8009 | 0.7978 | 0.8054 | 0.7995 | 0.8009 | 0.0028 |
| 74.81% | 0.8023 | 0.8011 | 0.7985 | 0.8062 | 0.7981 | 0.8012 | 0.0033 |
| 76.17% | 0.8046 | 0.7982 | 0.7990 | 0.8068 | 0.8012 | 0.8020 | 0.0036 |
| 77.46% | 0.8033 | 0.7988 | 0.7997 | 0.8047 | 0.8011 | 0.8015 | 0.0025 |
| 78.69% | 0.8037 | 0.7967 | 0.8012 | 0.8043 | 0.8012 | 0.8014 | 0.0030 |
| 79.86% | 0.8043 | 0.7967 | 0.8028 | 0.8062 | 0.7796 | 0.8019 | 0.0038 |
| 80.98% | 0.8035 | 0.7982 | 0.8030 | 0.8052 | 0.8001 | 0.8020 | 0.0028 |

Table 3.3: 5-fold cross validation of Active RNNs Classifier with Random Sampling

## 3.3 Active RNNs model with Uncertainty Sampling

Before showing the results, including table and figure, of active RNNs model with uncertainty sampling, we should clearly explain that:

(1) The following Table 3.4 and Figure 3.4 has same construct as the table and figure in last section. The filled data are different because of different sampling methods, i.e., in section 3.3, active RNNs with random sampling applies random sampling in selecting samples step of active learning, while active RNNs with uncertainty sampling exploits uncertainty (modified entropy) as an criterion for selecting samples.

(2) The goal of designing this experiment is verifying whether active learning with sampling strategy instead of random samling could actually reduce the size of required labeled data. Besides, as mentioned in section 2.3.1, uncertainty sampling is a conventional sampling strategy in active learning. We select it as a baseline for testing the efficiency of our proposed novel sampling strategy based on deep learning architecture.

(3) Table 3.4 reflects the details of 5-fold cross validation and Figure 3.4 summarizes 5-fold cross validation of active RNNs with uncertainty sampling.

According to Figure 3.4, the classification accuracy of active RNNs classifier with uncertainty sampling does approach the acceptable range $[81.61\%, 100\%]$ after employing $49.25\%$ of the whole data set, which is less than $80\%$. Unfortunately, it does not achieve the average line, i.e., $82.61\%$ before applying $80\%$ of the whole data set.

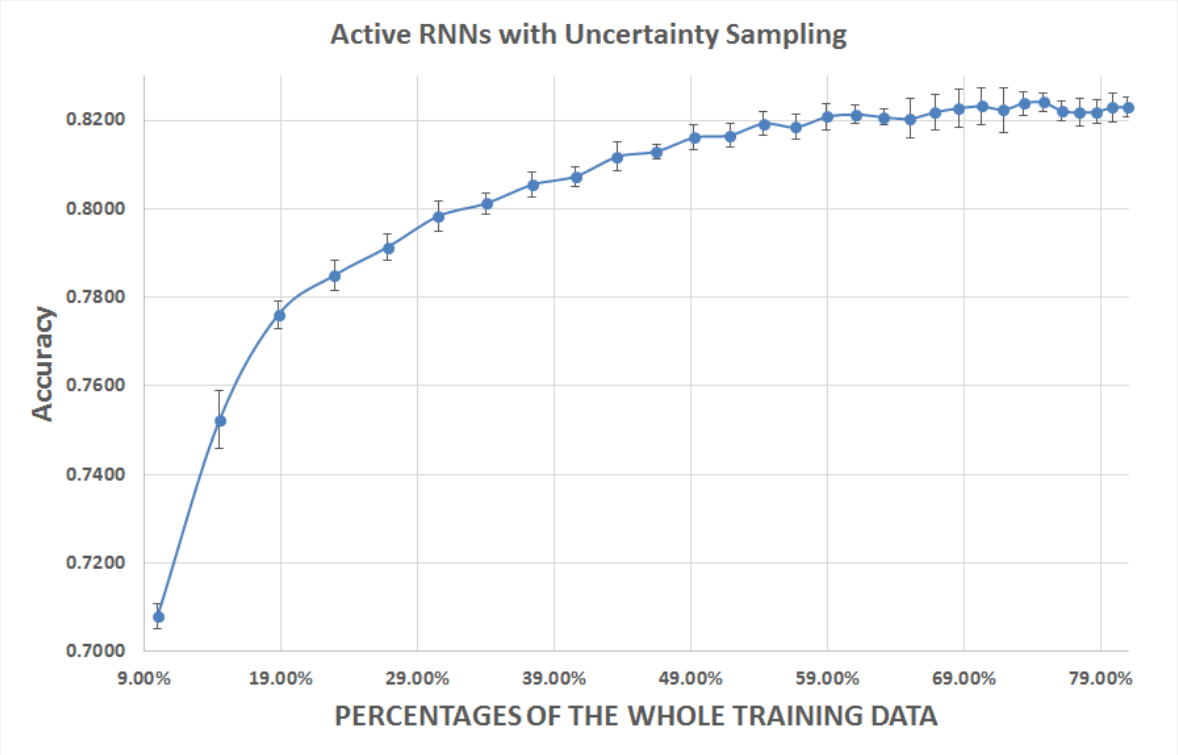


Figure 3.4: Active RNNs Classifier with Uncertainty Sampling

| Data Used | Test Accuracy (Group 1) | Test Accuracy (Group 2) | Test Accuracy (Group 3) | Test Accuracy (Group 4) | Test Accuracy (Group 5) | Average Test Accuracy | Standard Deviation |
|---|---|---|---|---|---|---|---|
| 10.00% | 0.7107 | 0.7041 | 0.7102 | 0.7089 | 0.7063 | 0.7080 | 0.0028 |
| 14.52% | 0.7553 | 0.7510 | 0.7503 | 0.7612 | 0.7437 | 0.7523 | 0.0065 |
| 18.83% | 0.7749 | 0.7745 | 0.7782 | 0.7803 | 0.7726 | 0.7761 | 0.0031 |
| 22.92% | 0.7813 | 0.7822 | 0.7883 | 0.7889 | 0.7842 | 0.7850 | 0.0035 |
| 26.81% | 0.7881 | 0.7889 | 0.7928 | 0.7954 | 0.7914 | 0.7913 | 0.0030 |
| 30.52% | 0.7956 | 0.7936 | 0.7995 | 0.8018 | 0.8011 | 0.7983 | 0.0036 |
| 34.04% | 0.7985 | 0.7987 | 0.8026 | 0.8027 | 0.8035 | 0.8012 | 0.0024 |
| 37.40% | 0.8103 | 0.8037 | 0.8034 | 0.8054 | 0.8044 | 0.8054 | 0.0028 |
| 40.58% | 0.8107 | 0.8059 | 0.8061 | 0.8086 | 0.8053 | 0.8073 | 0.0023 |
| 43.62% | 0.8161 | 0.8070 | 0.8124 | 0.8124 | 0.8108 | 0.8117 | 0.0033 |
| 46.50% | 0.8153 | 0.8122 | 0.8125 | 0.8137 | 0.8110 | 0.8129 | 0.0016 |
| 49.25% | 0.8208 | 0.8146 | 0.8140 | 0.8165 | 0.8145 | 0.8161 | 0.0028 |
| 51.86% | 0.8196 | 0.8143 | 0.8140 | 0.8188 | 0.8162 | 0.8166 | 0.0026 |
| 54.34% | 0.8201 | 0.8149 | 0.8202 | 0.8223 | 0.8187 | 0.8192 | 0.0027 |
| 56.70% | 0.8157 | 0.8153 | 0.8196 | 0.8215 | 0.8201 | 0.8184 | 0.0028 |
| 58.95% | 0.8197 | 0.8165 | 0.8210 | 0.8248 | 0.8218 | 0.8208 | 0.0030 |
| 61.08% | 0.8223 | 0.8179 | 0.8206 | 0.8235 | 0.8222 | 0.8213 | 0.0022 |
| 63.11% | 0.8223 | 0.8177 | 0.8205 | 0.8214 | 0.8215 | 0.8207 | 0.0018 |
| 65.04% | 0.8210 | 0.8206 | 0.8201 | 0.8263 | 0.8138 | 0.8204 | 0.0044 |
| 66.88% | 0.8229 | 0.8204 | 0.8212 | 0.8279 | 0.8166 | 0.8218 | 0.0041 |
| 68.63% | 0.8238 | 0.8171 | 0.8216 | 0.8292 | 0.8219 | 0.8227 | 0.0044 |
| 70.29% | 0.8251 | 0.8196 | 0.8197 | 0.8296 | 0.8217 | 0.8231 | 0.0042 |
| 71.87% | 0.8254 | 0.8175 | 0.8218 | 0.8293 | 0.8178 | 0.8224 | 0.0051 |
| 73.38% | 0.8226 | 0.8236 | 0.8217 | 0.8285 | 0.8229 | 0.8239 | 0.0027 |
| 74.81% | 0.8232 | 0.8238 | 0.8231 | 0.8277 | 0.8224 | 0.8240 | 0.0021 |
| 76.17% | 0.8257 | 0.8209 | 0.8217 | 0.8201 | 0.8221 | 0.8221 | 0.0022 |
| 77.46% | 0.8268 | 0.8215 | 0.8221 | 0.8194 | 0.8190 | 0.8218 | 0.0031 |
| 78.69% | 0.8241 | 0.8194 | 0.8228 | 0.8245 | 0.8186 | 0.8219 | 0.0027 |
| 79.86% | 0.8252 | 0.8194 | 0.8235 | 0.8267 | 0.8197 | 0.8229 | 0.0033 |
| 80.98% | 0.8244 | 0.8209 | 0.8209 | 0.8259 | 0.8223 | 0.8229 | 0.0022 |

Table 3.4: Active RNNs Classifier with Uncertainty Sampling 5-fold cross validation

## 3.4 Active RNNs model with Weighted Uncertainty Sampling

In this section, we present the results of experiment on active RNNs with weighed uncertainty sampling, which is the proposed novel algorithm in this project. The structure of Table 3.5 is same as Table 3.3 and Table 3.4. This experiment aims to verify the efficiency of the novel algorithm in deep learning application circumstances. As discussed in section 2.3.3, the criterion of selecting samples is called "weighted uncertainy". It is generated through uncertainty and representativeness, and representativeness is computed based on features extracted through deep learning.

Therefore, this experiment is the key of this project. It combines the idea of active learning and the methods of deep learing. If the results of this experiment confirms the efficiency of "deep active learning", there is no denying that active learing lower the threshold for using deep learning, and consequently overspread the application situation of deep learning. Because, in most cases, the labeled data is difficult to gather.

According to Figure 3.5, the classification accuracy of active RNNs classifier with weighted uncertainty sampling does approach the acceptable range $[81.61\%, 100\%]$ after employing $40.58\%$ of the whole data set, which is less than $80\%$. Besides, it reaches test accuracy $82.60\%$ when explioting $74.81\%$ of the whole data set and it almost achieves the average line $82.61\%$.

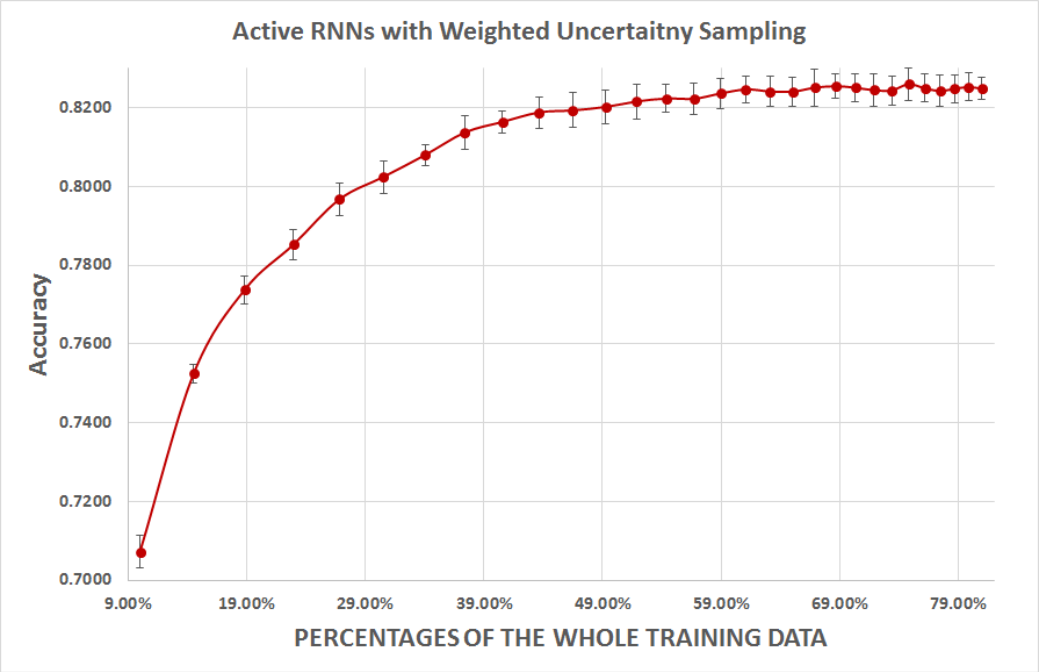


Figure 3.5: Active RNNs Classifier with Weighted Uncertainty Sampling

| Data Used | Test Accuracy (Group 1) | Test Accuracy (Group 2) | Test Accuracy (Group 3) | Test Accuracy (Group 4) | Test Accuracy (Group 5) | Average Test Accuracy | Standard Deviation |
|---|---|---|---|---|---|---|---|
| 10.00% | 0.7089 | 0.7026 | 0.7028 | 0.7114 | 0.7103 | 0.7072 | 0.0042 |
| 14.52% | 0.7537 | 0.7491 | 0.7522 | 0.7557 | 0.7514 | 0.7524 | 0.0025 |
| 18.83% | 0.7714 | 0.7706 | 0.7744 | 0.7796 | 0.7729 | 0.7738 | 0.0036 |
| 22.92% | 0.7828 | 0.7838 | 0.7845 | 0.7920 | 0.7830 | 0.7852 | 0.0039 |
| 26.81% | 0.7947 | 0.7938 | 0.7993 | 0.8028 | 0.7931 | 0.7967 | 0.0042 |
| 30.52% | 0.7984 | 0.8027 | 0.8012 | 0.8093 | 0.8007 | 0.8025 | 0.0041 |
| 34.04% | 0.8051 | 0.8062 | 0.8075 | 0.8119 | 0.8093 | 0.8080 | 0.0027 |
| 37.40% | 0.8069 | 0.8134 | 0.8137 | 0.8181 | 0.8166 | 0.8137 | 0.0043 |
| 40.58% | 0.8158 | 0.8157 | 0.8130 | 0.8209 | 0.8167 | 0.8164 | 0.0029 |
| 43.62% | 0.8191 | 0.8179 | 0.8133 | 0.8244 | 0.8194 | 0.8188 | 0.0040 |
| 46.50% | 0.8167 | 0.8191 | 0.8151 | 0.8267 | 0.8191 | 0.8193 | 0.0045 |
| 49.25% | 0.8178 | 0.8191 | 0.8160 | 0.8269 | 0.8213 | 0.8202 | 0.0042 |
| 51.86% | 0.8178 | 0.8198 | 0.8188 | 0.8289 | 0.8227 | 0.8216 | 0.0045 |
| 54.34% | 0.8201 | 0.8193 | 0.8220 | 0.8284 | 0.8219 | 0.8223 | 0.0036 |
| 56.70% | 0.8181 | 0.8224 | 0.8199 | 0.8288 | 0.8223 | 0.8223 | 0.0041 |
| 58.95% | 0.8191 | 0.8212 | 0.8237 | 0.8291 | 0.8250 | 0.8236 | 0.0038 |
| 61.08% | 0.8246 | 0.8225 | 0.8208 | 0.8300 | 0.8254 | 0.8247 | 0.0035 |
| 63.11% | 0.8231 | 0.8248 | 0.8184 | 0.8294 | 0.8251 | 0.8242 | 0.0040 |
| 65.04% | 0.8235 | 0.8218 | 0.8220 | 0.8307 | 0.8225 | 0.8241 | 0.0038 |
| 66.88% | 0.8283 | 0.8210 | 0.8210 | 0.8316 | 0.8236 | 0.8251 | 0.0047 |
| 68.63% | 0.8280 | 0.8227 | 0.8225 | 0.8296 | 0.8246 | 0.8255 | 0.0032 |
| 70.29% | 0.8269 | 0.8203 | 0.8229 | 0.8293 | 0.8261 | 0.8251 | 0.0035 |
| 71.87% | 0.8257 | 0.8218 | 0.8214 | 0.8312 | 0.8223 | 0.8245 | 0.0041 |
| 73.38% | 0.8250 | 0.8216 | 0.8206 | 0.8301 | 0.8247 | 0.8244 | 0.0037 |
| 74.81% | 0.8278 | 0.8228 | 0.8241 | 0.8325 | 0.8229 | 0.8260 | 0.0042 |
| 76.17% | 0.8276 | 0.8207 | 0.8227 | 0.8293 | 0.8244 | 0.8249 | 0.0035 |
| 77.46% | 0.8244 | 0.8215 | 0.8221 | 0.8311 | 0.8223 | 0.8243 | 0.0040 |
| 78.69% | 0.8269 | 0.8220 | 0.8216 | 0.8300 | 0.8237 | 0.8248 | 0.0036 |
| 79.86% | 0.8250 | 0.8230 | 0.8223 | 0.8311 | 0.8250 | 0.8253 | 0.0035 |
| 80.98% | 0.8262 | 0.8218 | 0.8242 | 0.8289 | 0.8231 | 0.8248 | 0.0028 |

Table 3.5: Active RNNs Classifier with WU Sampling 5-fold cross validation

# Chapter 4

# Analyses and Conclusion

In this chapter, we will analyze the above results from designed experiments and make a final conclusion.

Before starting the analysis, we should combine the above results into Figure 4.1. More specifically, there are 3 curves, represent active RNNs with random sampling, active RNNs with uncertainty sampling and active RNNs with weighted uncertainty sampling from the bottom up, and 2 horizontal lines indicate the average test accuracy of original RNNs classifier trained with the whole data and the test accuracy satisfied minimum requirement, from the top down. Besides, Figure 4.2 clearly reflects the relationship between different models.

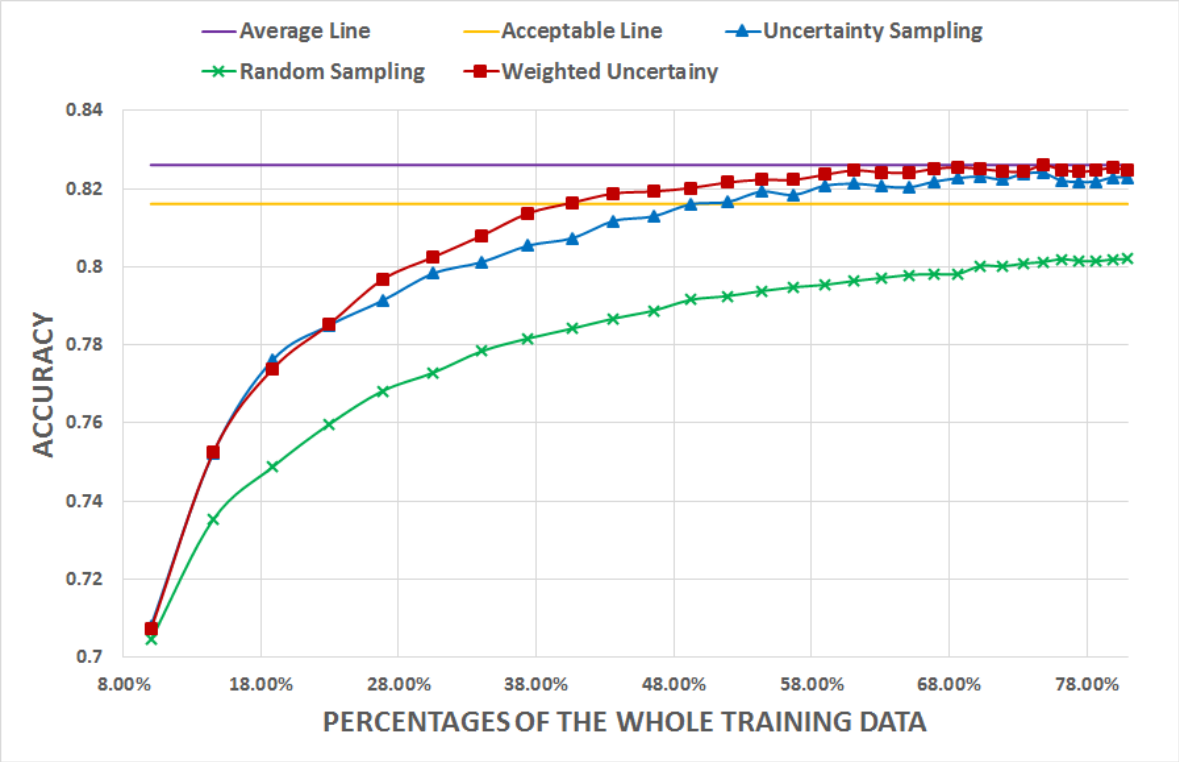


Figure 4.1: Weighted Uncertainty Sampling v.s. Uncertainty Sampling v.s. Random Sampling v.s. Acceptable Line and Average Line

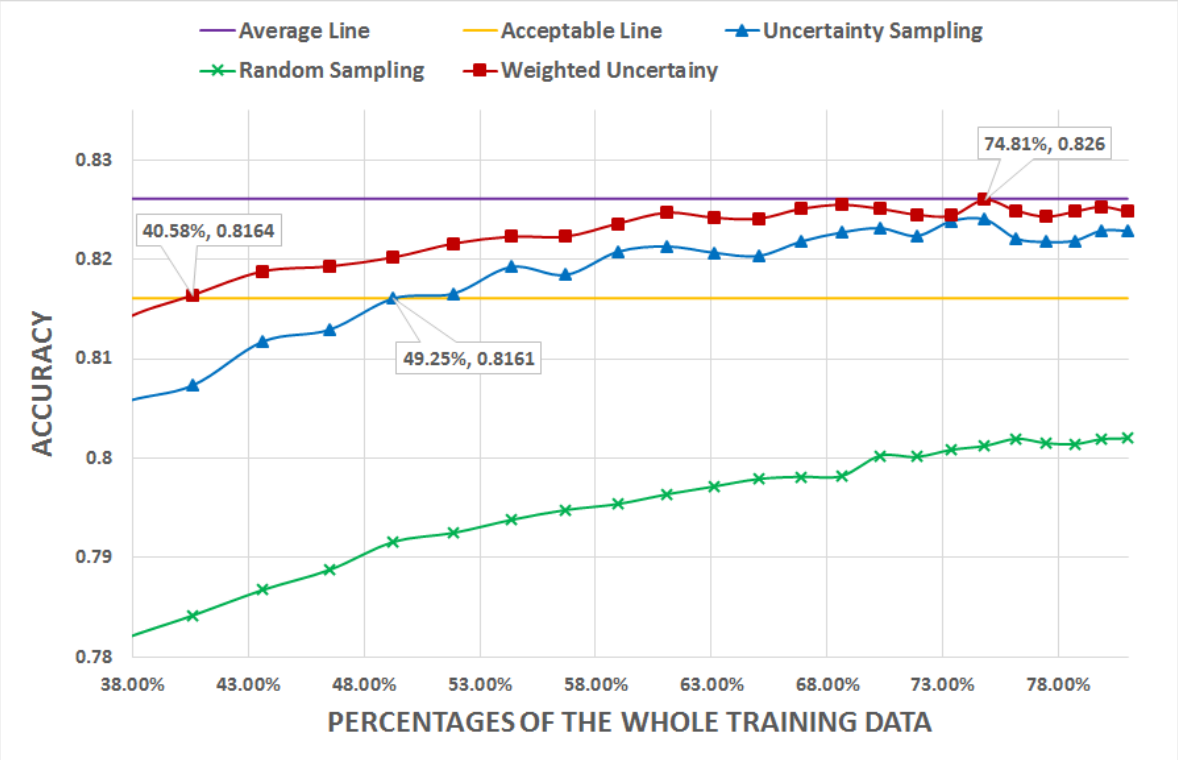


Figure 4.2: Weighted Uncertainty Sampling v.s. Uncertainty Sampling v.s. Random Sampling v.s. Acceptable Line and Average Line

**Note:** Active RNNs with weighted uncertainty sampling does not achieve the average line ($82.61\%$), but it almost approach it with test accuracy $82.60\%$ and $74.81\%$ of the whole training data. If we increase the size of training data set, the test accuracy of active RNNs with weighted uncertainty sampling might reach the average line before using the whole trainind data.

## 4.1 Analyses

The following paragraphes are organized on basis of results in above sections, especially Figure 4.1. The analyses are ordered from original RNNs classifier to active RNNs classifier with novel algorithm.

Firstly, we discuss the function of original RNNs classifier. The goal of designing 5-flod cross validation of original model is setting a baseline for verfying whether active learning make sense. Only when we know the performance of original RNNs classifier, we can compare it with performance of active learning and study the research question "To what extent can the amount of model data be reduced through active learning without decreasing the classification accuracy?". At present, we get the baseline of classification accuracy, i.e., $[81.61\%, 100\%]$, which is the orange line in Figure 4.1. If active learning exploits less than $80\%$ of the whole training data and achieves above classification accuracy, we can reach the conclusion that active learning can reduce the training data in an acceptable way.

Besides, we are eager to get the result that active RNNs classifier with proposed algorithm achieve the original test accuracy, i.e., $82.61\%$ while reducing the size of training

data. So, we keep the original test accuracy line in Figure 4.1, which is purple.

Secondly, we introduce the function of active RNNs classifier with random sampling. With the growth of the size of traing data set, the classification accuracy increase. This is a general phenomenon in machine learning area. Active learning with random sampling represents this process. Consequently, it is the baseline for active learning with other sampling methods. If one sampling method performs better than random sampling with same size of traing data, it indicates that this sampling method overcomes the influence of increasement of training data size. More specifically, the increase of test accuracy does not only rely on the increasement of training data, and relies on the sampling method. In conclusion, the goal of designing 5-flod cross validation of ative RNNs model with random sampling is setting a baseline of verfying whether active learning with other sampling methods are efficient. According the figure showed in section 3.5, random sampling cannot reduce the size of training data in an acceptable way and both weighted uncertainty sampling and uncertainy (entropy) sampling are efficient methods for sampling informative samples than random sampling.

Moreover, if we keep increasing the size of training data set to the size of the whole training data set, the test accuracy will be similar with test accuracy of RNNs classifier. Considering the influence of randomness, we use word $similar$ instead of $same$.

Thirdly, we present the function of active RNNs classifier with uncertainty sampling. According to Figure 4.1, the blue curve is always above green curve and exceeds the orange line when the training data set is around $49.25\%$, less than $80\%$. In other word, uncertainty sampling illustrates that it is more efficient than random sampling and can approach the research goal, reducing the amount of training data set without significantly influencing the test accuracy. The goal of designing 5-fold cross validation for active RNNs model with uncertianty sampling is building up a comparison for proposed novel algorithm, weighted uncertainty sampling. Also, if we didn't propose a novel algorithm, uncertainty sampling is also enough for us to achieve the research goal.

So, the application of uncertainty sampling as a representation of traditional active learning methods have two functions: (1) supporting a method to reduce training data set; (2) acting as a reference to verify the efficiency of novel algorithm. In following paragraphs we will focus on the performance of novel algorithm compared with uncertainty sampling.

Fourthly, we exhibit the performance of active RNNs classifier with weighted uncertainty sampling. Weighted uncertainty sampling considers both uncertainty and representativeness of a sample. Intuitively and theoritically, it selects more useful samples than uncertainty sampling. The improvement of weighted uncertainty sampling is that it considers the distribution of samples. In other words, each class should have a cluster center in feature space based on current model, and the situation is showed in Figure 2.2. We should think of the clusters. A sample closed to the center of one cluster is more important than an outlier sample. As showed in Figure 4.1, in most cases, the red curve is above blue curve. The result supports our hypothesis that weighted uncertainty sampling is better than uncertainty sampling and better than random sampling. Moreover, the red curve exceeds the orange line when the training data set is around $40.58\%$, less than $80\%$. So, the combination of deep learning and active learning is efficient for decreasing the required labeled data. Besides, the red curve approaches the purple average line when the training data set is $74.81\%$, so combining deep learning and active learning can get similar classification accuracy with less data than original RNNs classifier.

But, weighted uncertainty sampling is not so significantly better than uncertainty, which is indicated in Figure 4.2. There are two main reasons to explain this phenomenon:

(1) The distribution of samples in feature space is not strictly in clustering form, i.e., sometimes the samples from same class can't cluster. For example, class $black$, which is indicated as black points in Figure 2.2, have outliers, such as sample $B$, from point of view of distribution of samples. But these outliers are certain to class $black$. As mentioned, according to proposed novel algorithm, we prefer to select sample $A$ instead of sample $B$. The conclusion is that such samples have higher probability to be misclassified based on proposed algorithm than uncertainty sampling. Since we have lower chance to ask their true label during the process of active learning based on our proposed algorithm than uncertainty sampling, although they are not real outliers. The reason is that our application situation is "text classification", and consequently the expressions of one same text are variant. Espically, the text data is not in written form, but in oral form. As a result, the varieties are more. More specifically, "The order is canceled by the customer" is same to "The customer canceled the order" in meaning. Thus, they will be classified into the same class by human. But the difference between these samples in feature space is large. Unforturnately, the features space is $1200$ dimensions, thus we can not visualize this phenomenon directly.

(2) The labels of samples generated from human are not $100\%$ correct, i.e., human misclassify or mislabel some samples. More specifically, if we are facing the situation showed in Figure 2.2, we make our decision to select sample $A$ base on weighted uncertainty sampling. Assuming that sample $A$ is misclassified into class $orange$, the decision boundary will be influenced seriously, and the next active learning epoch for selecting samples will be forced to focus more on class $black$ since decision boundary is close to the center of class "black". Based on weighted uncertainty, we choose samples both close to boundary and close to evaluated class center. In other words, weighted uncertainty sampling is more sensitive to mislabel by human than uncertainty sampling.

So, the weighted uncertainty algorithm is not so significantly better than uncertainty sampling. But it is still a little bit more efficient than uncertainty sampling.

Until now, we have two aspect to improve in chapter further work based on above analyses. Firstly, we should make the text data samples in clustering form as much as possible through some deep learning method for features extracting. Compared with images, text data is harder to cluster or to be extracted features. Because, although the meaning of sentences is same, the expression format is totally different. In other words, the labels of text data samples are same, but their features are totally not relative in input space or features space. Moreover, the overlap of text data between different classes are more serious than images. When it comes to images, cats are cats, i.e., they have common features to be extracted easily even though cats show up in different environments. This phenomenon also indicates that people can more easily express their meaning through using images than just using text. Secondly, human annotators are not oracle in real world, thus they can make mistakes. The label of text samples are not $100\%$ correct. Also, different annotator has different preferences during the labeling process. Sometimes, this kind of preferences are changeable while they are labeling samples. As a result, this kind of noised data in industry is not as good as the data in academia, we should pay more attention to denosie text samples and focus on gathering true labels through some methods.

Above all, we have achieved the goal of research "reduce the amount of training data

without significantly influence the classification accuracy", and "our proposed algorithm based on RNNs architecture, i.e., active learning combines deep learning is better than just using uncertainty sampling".

## 4.2 Conclusion

In this section, we summary above analyses into brief descriptions based on our research objective.

In this project, we proposed a samples selecting methods in active learning based on RNNs architecture. The samples selecting process of candidate samples relies on representativeness and uncertainty, which is inspired by Density-Weighting Method [9]. The novel active learning algorithm, weighted uncertainty algorithm, which is suitable for searching samples in short-text situation, is developed by maximizing the weighted uncertainty function. Furthermore, this algorithm can be applied in any other deep learning architecture for classification task since the representiveness is calculated through features and deep learning is expert in features learning. According to the above results showed in Figure 4.1, the performance of the weighted uncertainty algorithm is better than two other methods, random sampling and uncertainty sampling. It is illustrated that the weighted uncertainty sampling algorithm has higher accuracy with less training data than other two methods. Besides, compared with original RNNs model, the active RNNs model with weighted uncertainty sampling can evidently reduce the amount of training data without signaficantly influence the classification accuracy. That could promote the widespread use of the deep learning since we can use less data than before. Using less data indicates that we decrease the cost of obtaining labeled data and overcome the bottleneck of labeling manually.

This project aims to overcome one of the bottlenecks of the deep learning in labeling data manually. In other words, labeling data limits the application of deep learning. If we break up the limitation of data, deep learning will perform better. Obviously, this project has a certain economic and social value. From the point of view of economy, active learning saves the money and the time for labeling data by human. From the point of view of society, active learning release the creativity of human being, who did the labeling job before. Besides, there is no ethics risk to combine active learning with deep learning. On the contrary, we benefit from this. Because, the work of labeling data is boring enough to efface the happiness of human. I have asked the colleagues, who label data in the company I served, and they are happy to hear that they can select less data than before. Furthermore, if we extend the application range of active learning in deep learning, the artificial intelligence industry will get benefits. And it have been proved that active learning is needed for some research domains which have few already labeled data for training a deep neural network [3].

# Chapter 5

# Future Work

In chapter 4, we have analyzed the results and give some explainations. Hence, in this chapter, we will discuss the potential room for improvement of the proposed algorithm in this project based on above analyses.

There are at least two further aspects can be modified for the purpose of improving the performance of proposed algorithm. The first one is that researching the distribution of samples in input space or features space and making the samples with same label into one cluster as much as possible. The second one is that labeling the selected samples as correct as possible or omitting the error labeled samples during training process based on some stragtegy, such as "cofident level of judging a mislabeled sample".

## 5.1 The Distribution of Samples

As mentioned, the distribution of samples is not strictly in clusters form. So, it has negative influence on the sampling process in active learning. In this project, we use the hidden vector generated from RNNs model of a sample as its features. Although the features vector can completely represents the sample, sometimes the similarity between two samples with same label is low in text situation.

In order to overcome the problem that samples with same label are not close to each other in feature space, we might try dimension reduction. Becasue from the point view of mathematics, lower dimension means samples with same label have higher chance to close to each other.

(1) We might try the output vector generated from RNNs model in the future. The output vector is compressed compared with hidden vector, i.e., output vector has lower dimension.

(2) The kernel idea of applying output vector is dimensionality reduction. Besides, conventional dimension reduction, such as Principal Component Analysis (PCA), of hidden vector might be considered.

However, dimension reduction brings overlapping between classes, and consequently it will be difficult for classification task. All in all, the kernel is making samples into cluster form in feature space. Under this circumstance, representativeness as a criterion for selecting informative samples can be more efficient. We might use other unsupervised learning method such as autoencoder for features learning.

## 5.2 The Error Reduction of Labels

Active learning methods, including uncertainty sampling and our novel algorithm, have been proved to sigificantly decrease the size of required labeled data samples to generate a classifier having similar classification accuracy as original RNNs classifier. However, active learning process demands oracle or human annotator, which can give the $100\%$ correct labels to samples. An empirical study for hand-written digit recognition [30] indicated that active learning performances badly with a human annotator [31]. Hence, as we analyzed in chapter 4, we should denoise the labels of samples in order to get the true label of samples as much as possible.

Currently, we are trying to eliminate the mislabeled samples through KNN [32]. When we gather the labels of samples, we seperate samples according to their labels into different class. In each class, we randomly select several samples as initial data. Then, we add samples into this data set according to KNN criterion. For example, if the candidate samples is labeled as class $black$ and majority of its $k$ nearest neighbors have label $black$, we add this sample into data set. If the label of candidate sample is disagree with majority of its $k$ nearest neighbors, we discard this sample. This process is called "KNN error reduction". Finally, the active RNNs classifier with weighted uncertainty sampling combined "KNN error reduction" can be illustrated in following Figure 5.1.

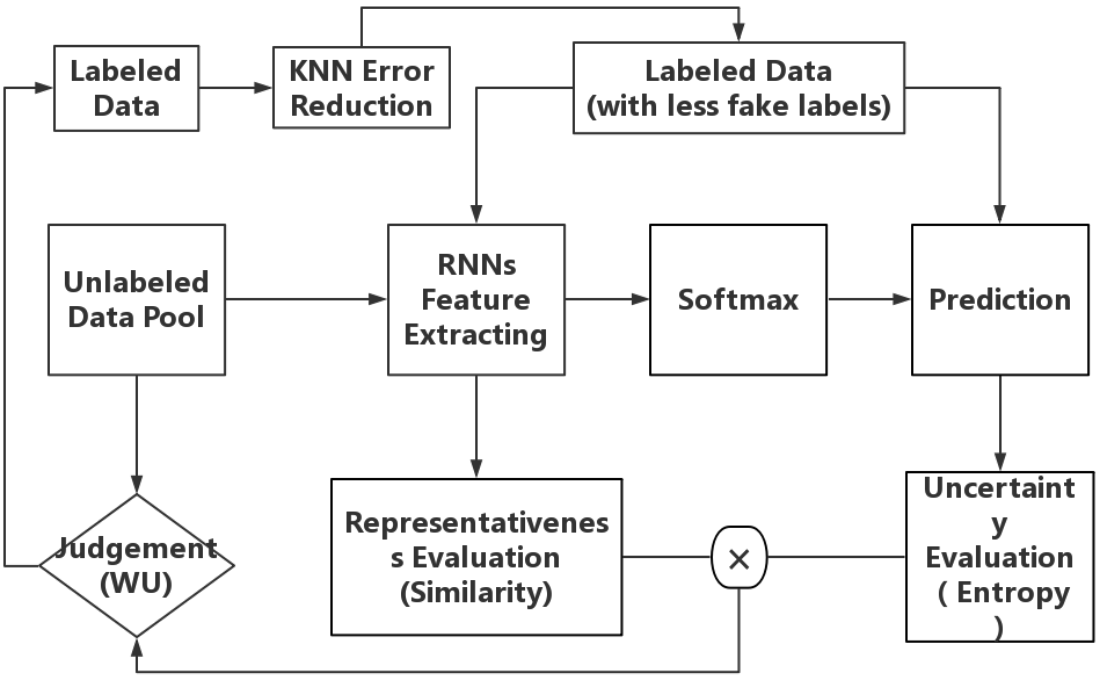


Figure 5.1: Active RNNs classifier with WU and error reduction

# Bibliography


[1] A. J. Joshi, F. Porikli, and N. Papanikolopoulos. Multi-class active learning for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2372–2379, 2009.

[2] Keze Wang, Dongyu Zhang, Y Li, Ruimao Zhang, and Liang Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 2016.

[3] Peng Liu, Hui Zhang, and K B Eom. Active deep learning for classification of hyperspectral images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, pages 1–13, 2016.

[4] Recurrent neural networks tutorial. `http://www.wildml.com`, .

[5] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[6] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, pages 3–12, New York, NY, USA, 1994. Springer-Verlag New York, Inc. ISBN 0-387-19889-X. URL `http://dl.acm.org/citation.cfm?id=188490.188495`.

[7] G. Tur and D. Hakkani. Combining active and semi-supervised learning for spoken language understanding. *Speech Communication*, 45(2):171–186, 2005.

[8] Burr Settles. Active learning literature survey. Coumputer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[9] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. *empirical methods in natural language processing*, pages 1070–1079, 2008.

[10] Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In *Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis*, IDA '01, pages 309–318, London, UK, UK, 2001. Springer-Verlag. ISBN 3-540-42581-0. URL `http://dl.acm.org/citation.cfm?id=647967.741626`.

[11] C E Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948.

[12] Seung, S H., Opper, and Sompolinsky. Query by committee. *Proc of the Fith Workshop on Computational Learning Theory*, 284:287–294, 1992.

[13] Kamal Nigam and Andrew McCallum. Employing em in pool-based active learning for text classification. In *Proceedings of ICML-98, 15th International Conference on Machine Learning*, 1998.

[14] David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine learning*, 15(2):201–221, 1994.

[15] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66, 2001.

[16] Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In *Advances in neural information processing systems*, pages 1289–1296, 2008.

[17] Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, volume 3, 2003.

[18] Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th international conference on Machine learning*, pages 208–215. ACM, 2008.

[19] Jingbo Zhu, Huizhen Wang, Tianshun Yao, and Benjamin K Tsou. Active learning with sampling by uncertainty and density for word sense disambiguation and text classification. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 1137–1144. Association for Computational Linguistics, 2008.

[20] Shusen Zhou, Qingcai Chen, and Xiaolong Wang. Active deep learning method for semi-supervised sentiment classification. *Neurocomputing*, 120:536–546, 2013.

[21] Andrew Kachites McCallumzy and Kamal Nigamy. Employing em and pool-based active learning for text classification. In *Proc. International Conference on Machine Learning (ICML)*, pages 359–367. Citeseer, 1998.

[22] Recurrent neural network. `https://en.wikipedia.org/wiki/Recurrent_neural_network`, .

[23] Andrew Ng and Jiquan Ngiam. UFLDL tutorial, softmax regression. `http://ufldl.stanford.edu/wiki/index.php/Softmax_Regression`.

[24] word2vec. `https://code.google.com/archive/p/word2vec/`.

[25] Jürgen Schmidhuber. Sepp hochreiter's fundamental deep learning problem(1991). `http://people.idsia.ch/~juergen/fundamentaldeeplearningproblem.html`, 2013.

[26] Junyoung Chung, Caglar Gulcehre, Kyung Hyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *Eprint Arxiv*, 2014.

[27] Peng Liu, Hui Zhang, and K B Eom. Active deep learning for classification of hyperspectral images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, pages 1–13, 2016.

[28] Sanjoy Dasgupta and John Langford. Active learning tutorial, icml 2009. `http://hunch.net/~active_learning/active_learning_icml09.pdf`.

[29] Andreas Krause. Advanced topics in machine learning, uncertainty sampling. `http://courses.cms.caltech.edu/cs253/slides/cs253-17-GP-BED.pdf`, 2010.

[30] Baum Eric B. and Kenneth Lang. Query learning can work poorly when a human oracle is used. *International Joint Conference on Neural Networks*, 8:8, 1992.

[31] Abraham Bernstein and Jiwen Li. From active towards interactive learning: using consideration information to improve labeling correctness. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, pages 40–43. ACM, 2009.

[32] Padraig Cunningham and Sarah Jane Delany. k-nearest neighbour classifiers. *Multiple Classifier Systems*, 34:1–17, 2007.

# Appendix A

# Appended Material

In this chapter, we introduces the resources and parameters we used in this project.

## A.1 List of the resources

(1) 73,747 labeled Chinese short-text samples in 435 classes.

(2) $65,447 \times 300$ $word2vec$ table, including $65,447$ words and phrases, and their corresponding $300$ dimension vectors.

(3) A cluster of high-performance GPU server with Nvidia K80 installed.

(4) Program Language: Python with RNNs TensorFlow 0.12.0.

## A.2 Parameters Tables

There are several parameters should be stated of RNNs classifier during the process of experiment. The RNNs in TensorFlow needs following parameters, and these parameters are generated from hundreds of experiments designed by the company. The are optimal parameters for this short-text classification task.

| Input Nodes | Hidden Nodes | Intial Range of $W, U, V$ | Dropout | Batch Size | word2vector Size | Number of Predicted Labels |
|---|---|---|---|---|---|---|
| 35 | 1200 | [-0.05, 0.05] | 0.35 | 20 | 300 | 10 |

Table A.1: Parameters for RNNs Classifier in TensorFlow

| Initial Learning Rate | Decay Epoch | Ratio | Number of Epoch | early stop |
|---|---|---|---|---|
| 0.500 | 10 | 0.9 | 39 | 10 |

Table A.2: Parameters for Training a RNNs Classifier

Besides, we designed some parameters during the experiments for the sake of saving computaional overhead and completely training the model. Some names of the parameters should be clearly decribed. In order to find the global extreme during training process, we should decrease the value of learning rate with the increase of training epoch.

But at the begining of training, large learning rate should be set for saving computaional overhead. Consequently, paramter $decayepoch$ means after this epoch, we will decrease our learning rate and $ratio$ means the decreasing ratio of learning rate.

Furthermore, the parameters in active learning process also should be listed. But we still trying to study the optimal parameter for active learning process in this project, for example, how much inital training data should we choose and in what kind of methods, random or clustering random.

| Initial Training Data | Increase Data Size of Labeling Step |
|---|---|
| 10% of the whole training data | 5% of the leftover unlabeled data |

Table A.3: Parameters in Active Learning