symmetry

MDPI

Article

Lifelong Machine Learning Architecture for Classification

Xianbin Hong [1,*], Sheng-Uei Guan [1], Ka Lok Man [1] and Prudence W. H. Wong [2]

[1] Research Institute of Big Data Analytics, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China; Steven.Guan@xjtlu.edu.cn (S.-U.G.); Ka.Man@xjtlu.edu.cn (K.L.M.)

[2] Department of Computer Science, The University of Liverpool, Liverpool L69 3BX, UK; PWong@liverpool.ac.uk

* Correspondence: Xianbin.Hong@xjtlu.edu.cn

Received: 11 March 2020; Accepted: 5 May 2020; Published: 22 May 2020

Abstract: Benefiting from the rapid development of big data and high-performance computing, more data is available and more tasks could be solved by machine learning now. Even so, it is still difficult to maximum the power of big data due to each dataset is isolated with others. Although open source datasets are available, algorithms' performance is asymmetric with the data volume. Hence, the AI community wishes to raise a symmetric continuous learning architecture which can automatically learn and adapt to different tasks. Such a learning architecture also is commonly called as lifelong machine learning (LML). This learning paradigm could manage the learning process and accumulate meta-knowledge by itself during learning different tasks. The meta-knowledge is shared among all tasks symmetrically to help them to improve performance. With the growth of meta-knowledge, the performance of each task is expected to be better and better. In order to demonstrate the application of lifelong machine learning, this paper proposed a novel and symmetric lifelong learning approach for sentiment classification as an example to show how it adapts different domains and keeps efficiency meanwhile.

Keywords: lifelong machine learning; continuous learning; sentiment classification; natural language processing

1. Introduction

Lifelong Machine Learning (or called as "lifelong learning" and "LML" for simple) is an advanced symmetric learning paradigm under the setting of continuous learning. The main aim of lifelong learning is to learn a new task more efficiently and effectively by learning procedure management and leveraging knowledge symmetrically [1–5]. From this view, lifelong learning is a great vision for the future and all of transfer learning [4,6], online learning [7,8], domain adaptation [9,10], reinforcement learning and etc. are the specific technologies for this vision. They have already achieved a milestone and are able to handle a set of similar tasks well now. To move forward, it is time to challenge more tasks or even infinite tasks.

In order to create a "Never-Ending" learning system [11], it is necessary to integrate isolated tasks and data together. Then all available resources include knowledge and algorithms should be allocated well for solving the incoming tasks. While the learning paradigm becoming more complicated, a management system should be set up for coordination. As previous research mainly study lifelong learning from a single angle, this paper will investigate what a complete lifelong learning system should contain and how it works.

The Section 2 will briefly introduce the lifelong learning and then Section 3 will express its knowledge system in details. A design of lifelong learning architecture and its implementation for sentiment

classification will be presented at Section 4 and a time complexity and a scalability analysis also will be provided. Then Section 5 will provide a discussion for the scalability of lifelong learning in big data processing and Internet of Things. Following, experiment results are going to be demonstrated at Section 6 and be discussed at Section 7. Finally, conclusions will be drawn at Section 9.

Contribution

This paper has following main contribution to science community:

1 Briefly reviewed the history of lifelong learning and discussed its relation to transfer learning.
2 Firstly summarised the two forms of knowledge: implicit knowledge and explicit knowledge and proved that creating a common data representation (implicit knowledge) is much effective for sentiment classification.
3 Set a baseline of BBSC model for user review sentiment classification.
4 Proposed a symmetric novel BBSC-KT lifelong learning model for sentiment classification and achieved a known state-of-art performance on Amazon dataset [12].
5 Proposed that Knowledge Assessor as a key component for lifelong learning.
6 Proposed to use the Lambda Architecture and NB-Student model for real-time predication, which has increased the scalability for lifelong learning.
7 Firstly discussed the scalability issues for lifelong learning in big data processing and IoT and suggested to involve edge computation.

2. Lifelong Machine Learning

Lifelong learning was proposed first at the end of last century [1,2], which aims for continuous learning. Previous researches mainly focus on the knowledge system for lifelong learning, try to leverage previous knowledge to new tasks. Transfer learning is a typical example [2,4]. According to [5,12], researchers may think lifelong learning is totally different with transfer learning. The authors ever also believed that and thought the difference between with them is that transfer learning is asymmetric and only transfers knowledge from only one source domain to an another domain while LML is symmetric and could transfer knowledge from various source domains to various target domains. However, they are only the same learning paradigm with different names. Tracing back to the earliest known paper of lifelong [2], an example of lifelong learning is to share knowledge among tasks by learning a general representation for images. This ideal is exactly same with creating a pre-trained model upon ImageNet [13] and which is the most famous case of transfer learning. According to the well-known survey of transfer learning [4], the term of "transfer learning" is raised at the same time with the "lifelong learning" by the same people, and they refer to same thing at that periods. From the points of view of some researchers [4,9], the learning approach of LSC [12] is also regarded as domain adaptation, which is a subset of transfer learning. Meanwhile, other publications refer to online learning [8] and memory reply [14], which are unable to explained by the definition of [5]. In summary, "lifelong learning" does not refer to any particular technology and it is a vision. It aims to re-use knowledge gathered in past to improve the the generalization accuracy [2] and many approaches include transfer learning, online learning and others could be leveraged.

In details, under the setting of lifelong learning, the system is not only learning to solve the current task but also learning the knowledge for past and future tasks, which is symmetric. When an algorithm tries to solve a new task, it could use both the knowledge generated from previous tasks and mines from itself. When mining knowledge from current task, some examples (labels) are necessary for supervised learning. Ideally, with more knowledge from previous tasks, less knowledge and less labels are required. Hence, using archived knowledge to help solving the new tasks with few labels is a natural way. This also is the foundation of the one shot learning or even zero shot learning.

Before achieving the goal of lifelong learning, the AI community would continuously has different understanding upon it. With the development of technologies, the approaches for lifelong learning is

changing. In current stage, to advance the transfer learning would be a reasonable way and this paper will demonstrate how it better than the ideal in [12].

2.1. Requirement for Lifelong Machine Learning Design

The reason why lifelong learning had been raised is because that the training cost for new tasks is expected to reduce, so the primary duty of it to reuse as much previous knowledge as possible and avoid unnecessary data annotation. In brief, the lifelong learning should be able to keep a high performance while providing less labelled data. This requirement would lead us to use more complex models such as large scale pre-trained neural networks. Such kind of models have excellent performance, but unfortunately, they are too large and too slow in many industry cases. Hence, we should to make sure the lifelong learning architecture keeping quick response while paying main attention to performance. In addition, knowing whether a new task is really a novel task or just a sub-task among solved problems is essential when dealing with incoming tasks. In summary, a lifelong learning architecture needs following characters:

(R1) Keeping high performance while lacking of labelled data by transferring knowledge
(R2) Training and prediction only cost reasonable time and other resources.
(R3) Being able to detect unseen tasks and propose possible solution.

Based on the goal of lifelong learning, above **(R1)** is obviously necessary. Considering there would be infinite tasks, **(R2)** also is crucial to make sure the scalability of LML. In addition, although most tasks have task segment [15], novel tasks would appear in real open world scenarios. Without knowing of such novel task, the lifelong learning is unable to determine how to change the strategies of knowledge transfer. Hence, **(R3)** also is important. Following Sections 2.1.1–2.1.3 will introduce how to satisfy above requirements.

2.1.1. Lifelong Learning for One-Shot Learning

As discussed above, the feature of one-shot learning [16] is most excepted to lifelong learning. To achieve this goal, knowledge transferring is the first choice. When talking about transfer learning, researchers mainly refer to the pre-trained neural network. Most famous one is the networks trained on the ImageNet [13]. Due to this kind of pre-trained models had been trained on a large scale of datasets by supervised or unsupervised learning, although they might have never seen a new task, it also can preform well upon some small datasets as they had already learned some essential skills. Hence, this research is determined to use pre-trained model to transfer implicit knowledge(please see Section 3.1 for details) from previous tasks to handle the new tasks with few data. With the feature of one-shot learning, the requirement **R1** could be met.

2.1.2. Lifelong Learning for Quick Response

Even though the pre-trained models had already replaced many classical classifiers such like KNN [17], SVM [18], etc. in several main stream application like image classification and video game, they have a serious problems with resource consuming. Large scale model might need days to train and could be also too slow while predicting. By this reason, such model couldn't to be deployed in real industrial environment. In order to solve this problem and deal with requirement **R2** and make sure high scalability, Knowledge Distilling [19] had became popular in recent years. Knowledge distilling raised a "Teacher-Student" learning paradigm, which means to use the large scale model to train a model as teacher at first and then use it to help small models with learning. Typically, this approach would extract the features of some specific layers from the large scale model and then transfer to the small models. These features contain rich knowledge which generated from large scale dataset. With knowledge distilling, small models can break the bottleneck of performance but keeping the scale. Inspired by it, lifelong learning architecture of this research would take a large pre-trained model as

"teacher" or "trainer" to help a small model with training and obtaining an impressive performance. In an industry situation, the pre-trained models are always trained for teaching the small model offline.

2.1.3. Lifelong Learning with Novel Task Detection

In lifelong learning, incoming tasks are various, some of them are similar with previous tasks while the others are totally new tasks which are difficult to deal with by current classifiers [20]. Hence, the lifelong architecture should be aware about it and provide a solution quickly. For this situation, this paper proposed to involve a novel task detection approach to determine whether a new task is an unseen task. If the new task is similar with any archived task, we will use just one or an ensemble of previous classifier to solve it. As for a novel task, a 'supervisor' will arrange a pre-trained model to learn it and then train a 'student' model.

2.1.4. Lifelong Architecture with Active Learning

When the train data is sufficient, the choice of classifier would only sightly influence the final performance. But when lacking of labelled data, algorithm becomes an important factor. If the new task is similar to previous tasks, normal transfer learning or domain adaptation would be helpful. However, lifelong learning always needs to face the novel tasks, which is difficult to deal with by only unsupervised learning or simple transfer learning. More labelled data is required, because even humans need new information for a new task. Considering there are already some previous knowledge could be reused, we do not need to label all the data. Instead of that, we need to choose the most valuable data for annotation. This raised the ideal of active learning [21], which means the system needs knowing when to ask human for help. In this research, we propose using clustering approaches to define the outliers in data and then label them.

2.2. Components of LML

The knowledge system contains two necessary components [12]:

- Knowledge-Base Learner (KBL): The KBL [12] responsible to discover knowledge from previous tasks and then transfer to the current or future task. Therefore, it includes two components: task knowledge miner TKM and task knowledge learner TKL. The miner discovers and evaluate where the knowledge is worth for reusing, while the learner extracts archived knowledge to another task.
- Knowledge Base (KB): The knowledge Base(Figure 1) responsible for maintaining the archived knowledge. It has two forms according to the knowledge type: Past Information and Meta-Knowledge.

Above two parts are necessary because knowledge mining and transfer are the main approaches to avoid learning from scratch. Knowledge here could have various forms and knowledge base also is different in different algorithms, please see Section 3 for details.

A lifelong learning system would have following advanced components, which are not easy to be implemented based on current technologies:

- Knowledge Assessor (KA): The knowledge Assessor is responsible to evaluate which previous knowledge could be re-used for a new target domains.
- Knowledge Reasoner (KR): The knowledge reasoner aims to regenerate more new knowledge based on the archived knowledge. The new knowledge could be meta-knowledge and logic inference might be needed.

Knowledge Assessor is a crucial component for the lifelong learning due to some knowledge might be unsuitable for all tasks. Like a word could have different meaning in different situation, a "previous knowledge" might not correct in a different scenario. The success of LSC [12] also benefited from this mechanism. It involved a concept of "domain knowledge", which is only validate for a

specific domain. The restriction upon knowledge transfer could minimize the side effect of improper knowledge transfer.

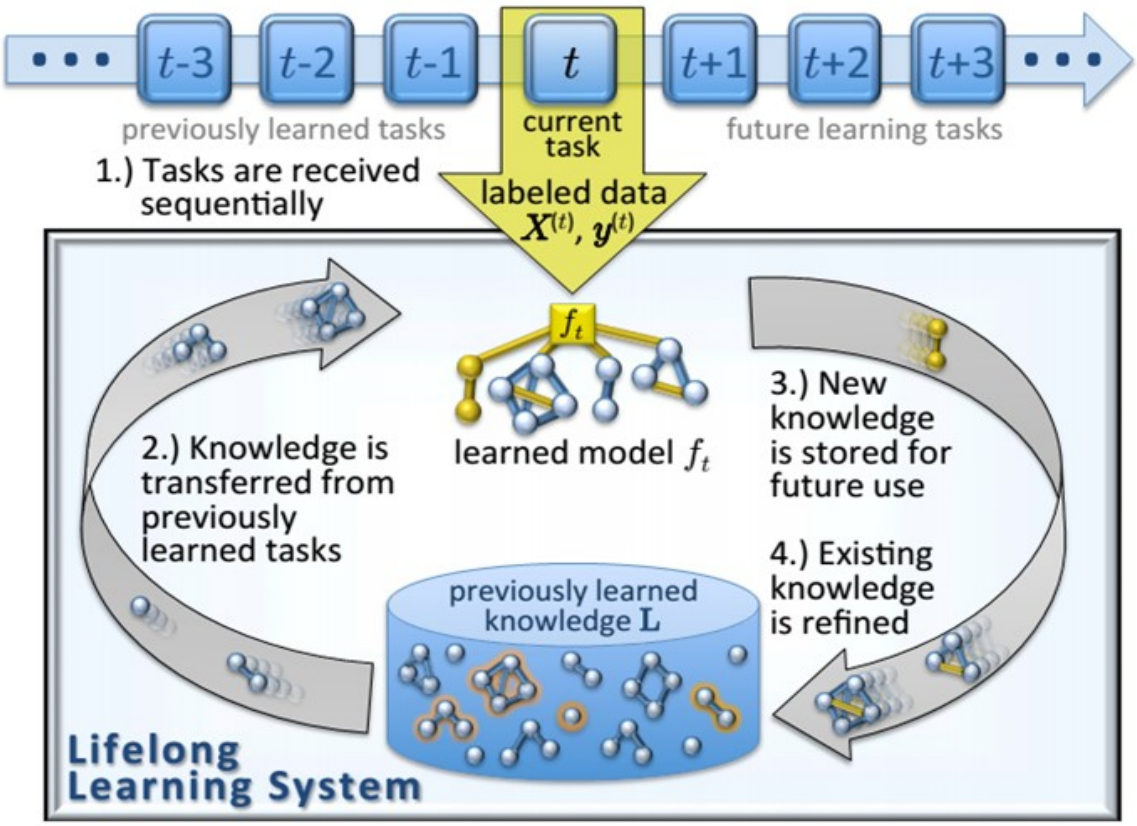

Figure 1. Knowledge System in the Lifelong Machine Learning [22].

2.3. Multi-Layer Lifelong Learning Architecture

From introduction Section 2.2, it is obvious that a lifelong architecture has many components and this paper roughly categorizes them into a front-end, middle-layer and a back-end. The front-end refers to the models which are dealing with tasks directly. They are normally the small models and will be called as 'student models' in the rest of this paper. Within the middle-layer, there is the large scale pre-trained models which have remarkable performance but consume significantly too. They will be called as 'trainer models' or 'teacher models'. These pre-trained models are responsible to mine and learn knowledge from tasks, as KBL metioned in Section 2.2. The most complex part is the back-end, which mainly contains a 'supervisor' or called 'monitor' as a manager for the whole system. It is responsible for arranging the 'trainer models' to teach the 'student models', to monitor knowledge storage and so on. Additional, back-end also involves the novel task detector, knowledge base, reasoner KR, knowledge assessor (KA) and other necessary components for lifelong learning. In brief, a lifelong architecture contains following components:

- Front-End: Classifiers who are directly handling the tasks.
- Middle-Layer: Pre-trained models who are responsible to train the front-end classifiers; KBL.
- Back-End: All the rest components supporting the front-end and middle-layer. Including:
  - Supervisor/Manager: The supervisor is responsible to supervise all the process within the system, assign tasks to other components.
  - Novel Task Detector: The detector needs to watch all the incoming tasks and try to recognize the novel tasks by similarity analysis.

- Knowledge Base KB: The component involves all the useful information generated from previous or current tasks, which includes previous data, knowledge and meta-knowledge.
- Knowledge Assessor KA: Crucial part to make sure all knowledge could be re-used properly.
- Meta-Knowledge Miner: This miner needs keeping discovering the meta-knowledge among tasks.

3. Knowledge

3.1. Implicit and Explicit Knowledge

As knowledge is the foundation of lifelong learning, the choice of knowledge will determine the architecture of lifelong learning.In order to help researches understand how to choose knowledge, this paper categories knowledge as and explicit form. Implicit knowledge means that it is not directly defined by human but exactly helpful for problem solving. Like the parameters of neural networks, they also could be transferred to a new task by transfer learning. Explicit knowledge normally can be directly defined and understand by human, like knowledge graph. Explicit knowledge has better interpretability and easy to modify. The choice of above two forms represent two direction for lifelong learning research. Researches use implicit form normally prefer to improve the representation for tasks, which could support different kinds of task. If using explicit form, the knowledge might only could be transferred in the same kind of tasks. For instance, the knowledge defined for sentiment classification [12] is hard to use for named entity recognition, but implicit form by BERT [23] has no this problem. This paper suggests researchers to pay main attention on creating a general representation as implicit knowledge. And the explicit knowledge only needs to design for specific task as supplement. Below section will prove that this way has better performance than LSC [12] and LNB [24], which are only working on explicit knowledge.

3.1.1. Implicit Knowledge

As implicit knowledge normally is the parameters of algorithms, it usually only can be reused by transfer learning. Typically like using pre-trained model firstly learned the representation from some large datasets and then fine-tine on target task. The learned knowledge is saved in the pre-trained model and helpful for the target task.

The main drawback of such implicit knowledge is that it is hard to modified when there are some specific requirements. For example, BERT (Bidirectional Encoder Representations Transformers) [23] has a very good and general representation for text, could understand the relationship between words. However, in some e-commerce tasks, the relationship between some named entities are important. Ideally, the named entities only a small amount comparing with the vocabulary that BERT ever learned, so it should be easy to learn. But in fact, it is not easy to directly teach BERT with the named entities. A possible way is to use named entities masking to retrain BERT, which with large resource consume.

The similar limitation also happened in vision field. A large network pre-trained upon ImageNet could accelerate the training of a new task and also improve the performance upon it. However, parameters transforming is the only way to apply the pre-trained model. It's impossible to directly tell the model to pay more attentions to which part of object and how to distinguish different object. Labeled data still is necessary, which means the pre-trained model lacks the ability to inference by itself.

Whatever in NLP or vision field, the usage of implicit knowledge is confined to transforming parameters from pre-trained model to accelerate new tasks' training. Although a large pre-trained model had already read huge amount of texts or seen various images, it still is unable to tell people how to understand a sentence or how to distinguish two objects. In such situation, a model is unable to evaluate how it confident to its predication, which is unacceptable in many scenarios like medical treatment and auto driving. Another problem is that a lifelong learning algorithm should know which knowledge is needed to solve a new task. It is supposed to learn such knowledge by itself as much as

possible. But if it doesn't know which knowledge dose the pre-trained model already contain, it is obviously impossible to self-study.

Overall, the implicit knowledge is helpful and could be gained by unsupervised learning which saves human labor a lot. But this kind of knowledge is hard to understand by human and so it is obviously difficult to let human guide the machine to apply the implicit knowledge well. There should be an interface between human and machine to let human can assist machine to make a good use of the implicit knowledge.

3.1.2. Explicit Knowledge

Explicit knowledge in natural language processing (NLP) mainly includes dictionary, knowledge graph and statistic probability and etc. For instance, researches use stop word dictionary to eliminate stop words, use knowledge graph to determine the relationship between objects and use statistic probability to predict next word in a sentence.

It is not easy to define explicit knowledge in some situation. Like in image recognition, we can tell the machine that a car has doors, but are unable to tell it how to know whether an object is a door. While human don't know how to solve a problem, we are also unable to provide explicit knowledge to the machine, which is a huge limitation.

Meanwhile, to provide the explicit knowledge always means manpower consuming. Explicit knowledge always is correct because most of them were checked by human but which also cost huge time. Though time has spent, it is more easy to build rulers and inference based on such correct and understandable knowledge.

In summary, explicit knowledge is more accurate, understandable and easy to apply while implicit knowledge contains plentiful knowledge which human haven't discovered. None of them could replace each other, the best way for human is to find an interface to integrate them and also discover more valuable information from implicit knowledge.

In summary, implicit form could support knowledge transfer among more tasks than explicit form, so it could gain more knowledge and be more powerful. As for explicit form, it is more suitable as a flexible supplement to improve the specific task.

3.2. Knowledge in Different Learning Phrase

Within lifelong learning, it is important to understand where the knowledge comes from. Before learning a new task, there is past knowledge which generated from previous tasks or directly provided by human intelligence. During learning, unsupervised learning may help to mine some new knowledge, which is meta knowledge. After learning, a model had generated and also contains knowledge which may contribute to later tasks.

3.2.1. Past Knowledge

The past knowledge includes the trained model or other useful information mined from previous tasks. As for lazing learning like K nearest neighbor algorithm [17], past data also is past knowledge. In addition, available explicit knowledge like knowledge graph also could be regarded as past knowledge. A knowledge validation is necessary here to determine which past knowledge is able to use in current task. Domain adaptation is one approach to help knowledge transform in a reasonable way.

3.2.2. Meta Knowledge

During current task learning, some extra knowledge could be extracted from data, which also called meta knowledge. Meta knowledge usually be regarded as new input features to help training. Such meta knowledge also should be stored in knowledge base to help further task solving.

3.2.3. Learning Outcome as Knowledge

After training, whatever supervised or unsupervised or reinforcement learning, the generated model also contains knowledge which generated from current task. The results or predictions also have knowledge. Hence, whatever the model itself or the prediction generated by it should be regarded as new knowledge.

3.3. Knowledge Validation

Knowledge validation responses to evaluate which kind knowledge is correct and should be stored and provided to further tasks. Although knowledge is important, some of them still could be harmful or improper for the current task. It is possible that the knowledge was gained from wrong conclusions, and even "correct" knowledge also is only validate in specific fields. Hence, knowledge validation is essential in lifelong learning. It exists at two time slots: before archiving knowledge to knowledge base or before retrieving.

Previous researches rarely paid enough attentions to validation as it's challenged, but it's really a crucial part of lifelong learning. At the system design stage, It is necessary to figure out which knowledge should be stored considering the capacity of the knowledge base. This because that it is impossible to store all information into knowledge base, only the most valuable part worth.

When retrieving knowledge from knowledge base, validation also is necessary. This because the target task absolutely has differences with the previous tasks, which include data distribution and task requirements.

Based on discussed above, knowledge validation raised high requirement for knowledge management. After knowledge generated, we need to filter them and only reserve those could be reused again. When entering the knowledge, many additional features or tags should be add to the knowledge. The tags can help us to determine which knowledge is suitable for current task. For instance, each word in sentiment classification will be assign a tags to indicate its usage area. The word "wonderful" is a positive word in all domains while "bright" might only be a positive word in some product review domains like "Flash Light". The tags will mark them and avoid to improper transfer. For this paper, each classifier has a tags to mark which task it is corresponding with. In lifelong learning, the knowledge assessor (KA) is the key component to improve the performance of the system.

4. Natural Language Processing with Lifelong Architecture

In practice, this paper will use the sentiment classification to demonstrate the lifelong learning architecture design. Hong and etc. [25] had argued that the NLP field is the best choice for the lifelong machine learning researches. Because natural language is easy to be understood by human and the knowledge also could be defined by human directly. All natural languages has basic component like character in Chinese and letter or word piece in English. Whichever NLP task shares such basic component so it's possible to build an interface between them to exchange knowledge. In practice, the NLP tasks include sentiment classification, named entity extraction, dialogue system, machine translation. These tasks usually are treated individually, but they definitely have intrinsic connection. For the machine translation task, knowledge of named entities could be helpful for translating the phrases about places, organizations and etc. As for named entity extraction, part of speech also helpful due to entities normally are noun. Hence, we believe that it is necessary to treat a target task with different learning approaches in parallel. This means that even for a sentiment classification task, part of speech analysis, named entity extraction and etc. also should be taken in order to discover knowledge as more as possible.

4.1. Sentiment Classification

Previous papers [12,20,24,25] chose the sentiment classification for lifelong learning researchers due to there are many related but different sub-tasks. They are related so the knowledge is possible to transfer. But they are also different, so the knowledge transfer also would be harmful. Therefore it is necessary to determine whether the archived knowledge is helpful or harmful. A real "lifelong learning" should solve this problem to ensure that its performance would be continuously improved.

Although deep learning had been widely used for NLP and sentiment classification, it still has limitation for lifelong learning. As deep learning is always regards as a "black box", it's difficult to define and reuse knowledge freely. In the previous work [12], researches used Naïve Bayes probability to represent knowledge. In this case, each word has a probability of occurrence in positive reviews or negative reviews. "Lifelong Sentiment Classification" ("LSC" for simple below) [12] not only uses these information within one domain, but also stores them in knowledge base for reuse. It will check whether a word always is positive or negative as meta-knowledge. With such across domain meta-knowledge, it could leverage past information to a new task.

4.2. Lifelong Architecture for Sentiment Classification

Based the lifelong architecture design mentioned above, we had raised a specific version for the sentiment classification:

- Front-End: Classical classifiers like Naïve Bayesian.
- Middle-Layer: Pre-trained model BERT [23]; fine-tuned BERT model to recognize negative reviews by transfer knowledge cross tasks.
- Back-End: All the rest components supporting the front-end and middle-layer. Including:
  - Supervisor/Manager
  - Novel Task Detector: A trained task-classifier to categorize incoming tasks. Previous classifiers providing assistance.
  - Knowledge Base: Storing the past data, past models.
  - Knowledge Miner:Extracting meta-knowledge from tasks, like named entities, sentiment words and etc., which would contribute to the further training.

For a new task, the 'supervisor' will firstly check whether it's a novel and unseen task as the Algorithm 1. There are two approaches to detect the novel task. One is to comparing the similarity on the text, the another one is trying to use the previous task's classifiers to predict the new task. If any reasonable prediction performance obtained, the new task could be treated as a known task [15,19]. In meanwhile, the knowledge base, knowledge miner also are working to provide supports.

From the view of another angle, the architecture also could be regarded as an online-offline system:

- Online: front-end prediction, knowledge extracting
- Offline: model training, novel task detection

If the new task had been determined as a novel task and there is not any previous classifier is able to handle it, the 'supervisor' will assign a pre-trained model to learn it based on Algorithms 2 and 3. If the labeled data is insufficient, active annotation (Algorithm 4) could help to maximum the effectiveness of limited data.

Algorithm 1: Novel Task Detection

Input: A new task, *similarityThreshold*: λ_1, *performanceThreshold*: λ_2

Output: *isNovelTask*: boolean, *relatedTasks*: related task ID, *classifiers*: suitable classifiers

1	*data* ← data of new task	$O(1)$
2	*documentEmbedding* ← *data*	$O(1)$
3	**for** *each previous task* T_i **do**	
4	document embedding $E_i \leftarrow T_i$	$O(i)$
5	task similarity $S_i \leftarrow$ comparing similarity between *documentEmbedding* and E_i	$O(i)$
6	**if** $S_i > \lambda_1$ **then**	
7	*relatedTasks* ← T_i	$O(i)$
8	**for** *each trained classifier* C_i **do**	
9	$P_i \leftarrow$ test C_i's performance on new task	$O(i)$
10	**if** $P_i > \lambda_2$ **then**	
11	*classifiers* ← C_i	$O(i)$
12	*relatedTasks* ← C_i's corresponding T_i	$O(i)$

Algorithm 2: BBSC Model Training

Input: A new task with N labelled data, pre-trained Bert, iteration number i

Output: BBSC Model

1	*trainData* ← Train Data	$O(1)$
2	*trainer* ← Pre-trained Bert Model	$O(1)$
3	*wordEmbedding*, *positionEmbedding*, *tokenEmbedding* ← *trainData*	$O(1)$
4	*sentenceEmbedding* ← *wordEmbedding*, *positionEmbedding*, *tokenEmbedding*	$O(1)$
5	Train *trainer* upon *sentenceEmbedding*	$O(i * N)$
6	Deploy *trainer* to back-end server	$O(1)$

Algorithm 3: New Task Training

Input: A new task with N labelled data, pre-trained Bert, iteration number i

Output: trainer model, student model

1	*trainData* ← Train Data	$O(1)$
2	*testData* ← Test Data	$O(1)$
3	*trainer* ← Pre-trained Bert Model	$O(1)$
4	Train *trainer* upon *trainData*	$O(i * N)$
5	$\hat{y} \leftarrow$ *trainer* predicts *testData*	$O(N)$
6	*pseudoLabel* ← $\hat{y}$	$O(1)$
7	*pseudoTrainData* ← *trainData*, *pseudoLabel*	$O(1)$
8	*student* ← Naïve Bayesian model	$O(1)$
9	Train *student* upon *pseudoTrainData*	$O(N)$
10	*Knowledge Base* ← *pseudoLabel*	$O(1)$

Algorithm 4: Active Annotation

Input: A novel task, *sampleNumber*: N_1, *outlierThreshold*: λ_3 *annotationNum*: N_2

Output: *annotationData*

novelData ← data of new task	$O(1)$
previousSamples ← Sample N_1 data from previous task	$O(1)$
Outliers← Clustering *previousSamples* and *novelData* with λ_3	$O(N_1 \log(N_1))$
C ← center of clustered data	$O(1)$
Sort *Outliers* by distance to C	$O(N_1 \log(N_1))$
Remove *previousSamples* from *Outliers*	$O(1)$
annotationInstances ← collect farthest N_2 instances from C in *Outliers*	$O(1)$
annotationData ← ask human to annotate *annotationInstances*	$O(N_2)$

4.3. Naïve Bayesian (NB) Classifier

Naïve Bayesian (NB) classifier [26] is a classical algorithm and widely used for sentiment classification.

$$P(w|c_j) = \frac{\lambda + N_{c_j,w}}{\lambda\,|V| + \sum_{v=1}^{V} N_{c_j,v}} \tag{1}$$

where $P(w|c_j)$ is the probability of a word occurs in a specific class, c_j is a single class. $N_{c_j,w}$ represents the word frequency in a class. And $|V|$ shows the vocabulary V size and $\lambda(0 \leq \lambda \leq 1)$ is smoothing factor.

The probability for a specific document is:

$$P(c_j|d_i) = \frac{P(c_j)\prod_{w\in d_i} P(w|c_j)^{n_w,d_i}}{\sum_{r=1}^{C} P(c_r)\prod_{w\in d_i} P(w|c_r)^{n_w,d_i}} \tag{2}$$

where d_i is the given document, n_w, d_i is the word frequence.

To predict a document, only $P(c_+|d_i) - P(c_-|d_i)$ is needed.

$$P(c_+|d_i) - P(c_+|d_i) = \frac{P(c_+)\prod_{w\in d_i} P(w|c_+)^{n_w,d_i}}{\sum_{r=1}^{C} P(c_r)\prod_{w\in d_i} P(w|c_r)^{n_w,d_i}} - \frac{P(c_-)\prod_{w\in d_i} P(w|c_-)^{n_w,d_i}}{\sum_{r=1}^{C} P(c_r)\prod_{w\in d_i} P(w|c_r)^{n_w,d_i}} \tag{3}$$

As only $P(c_+|d_i) - P(c_-|d_i)$ is matter, above formula could be change to:

$$P(c_+|d_i) - P(c_+|d_i) = P(c_+)\prod_{w\in d_i} P(w|c_+)^{n_w,d_i} - P(c_-)\prod_{w\in d_i} P(w|c_-)^{n_w,d_i} \tag{4}$$

Here, a word's sentiment polarity is determined by probability of it occurs in each class. And the sentiment polarity of each word will determine the polarity of a document together.

4.4. Baseline

This paper will use the most famous **lifelong learning sentiment classification-LSC** [12] and its newest updated version **"Lifelong Naïve Bayes" LNB** [24] as baselines. LSC only transfers knowledge from previous tasks to new task while LNB also transfers back.

4.5. BERT Based Model for Sentiment Classification

BERT (Bidirectional Encoder Representations from Transformers) [23] is a technique for NLP pre-training and developed by Google. By the unsupervised pre-training, BERT had built an outstanding symmetric language model, which use self-attention transformer (see Figure 2 as reference). It could transform a sentence to a word vector, a numeric representation. When use it for a specific task, there are following steps:

1. Adding a new layer/network upon top of the BERT output.
2. Calculating the probability for each document.

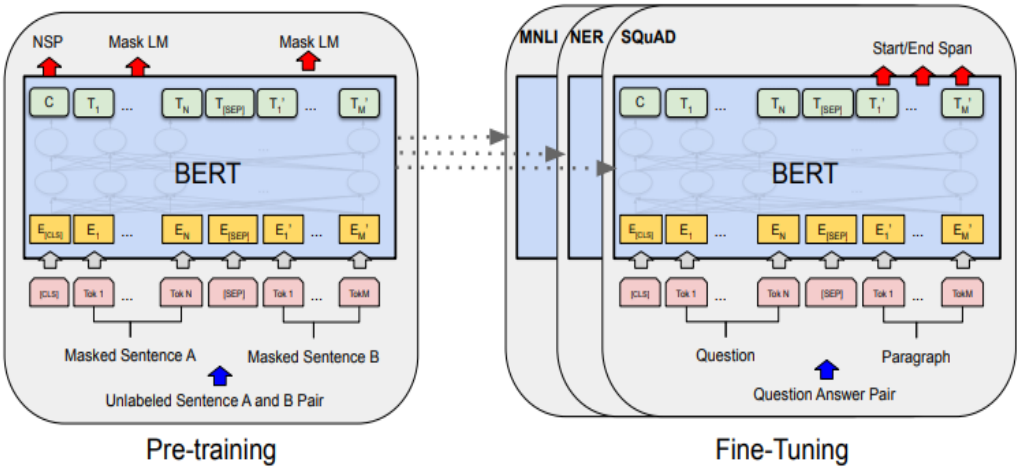

Figure 2. BERT Model Structure [23].

BERT also use the transfer learning technique. During the unsupervised pre-training, the neural network had learned the relationship between words. Using this pre-trained model with fine-tine in down-stream tasks, the past knowledge could transfer to current task. For simple, this paper will call the **BERT based down-stream Sentiment Classification Model** as **BBSC model**. Although BERT based model can achieve the state-of-art performances upon many tasks, it's not a continuous learning design. A fine-tuned model normally is inappropriate to use for another task. Hence, we would leverage the pre-trained BERT model to train upon each tasks to deliver different BBSC Model.

4.6. BBSC with Knowledge Transfer

Considering the main problem in sentiment classification is due to the asymmetric and imbalanced data distribution. As the number of positive review is around 5 times than negative review, most classifiers lack of knowledge of negative class. To solve this problem, it is nature to consider to transfer knowledge from other domains. However, each BBSC model is only designed for a specific domain and unable to use for another domain(Section 6.6 will indicate that BBSC will get lower performance on other domains). Even it is possible to use a single BBSC model to learn all tasks, the forgetting problem also is hard to avoid [14]. BBSC is a neural network, when learning upon new domains, the old parameters will be replace and cause to reduce the performance on older domains. Hence, this paper proposes a novel knowledge transfer approach-BBSC model with knowledge transfer(BBSC-KT) to improve original BBSC model (Figure 3).

The main ideal of BBSC-KT is only leverage necessary knowledge rather than all knowledge. For this sentiment task, each domain has enough positive review so they are unnecessary. As mentioned before, BBSC is suffering from lacking of negative review, so BBSC-KT will focus on knowledge transfer upon negative reviews. Also, BBSC-KT transfer knowledge from similar domains, this based on the measurement of task similarity(see Algorithm 1 and Section 6.9).

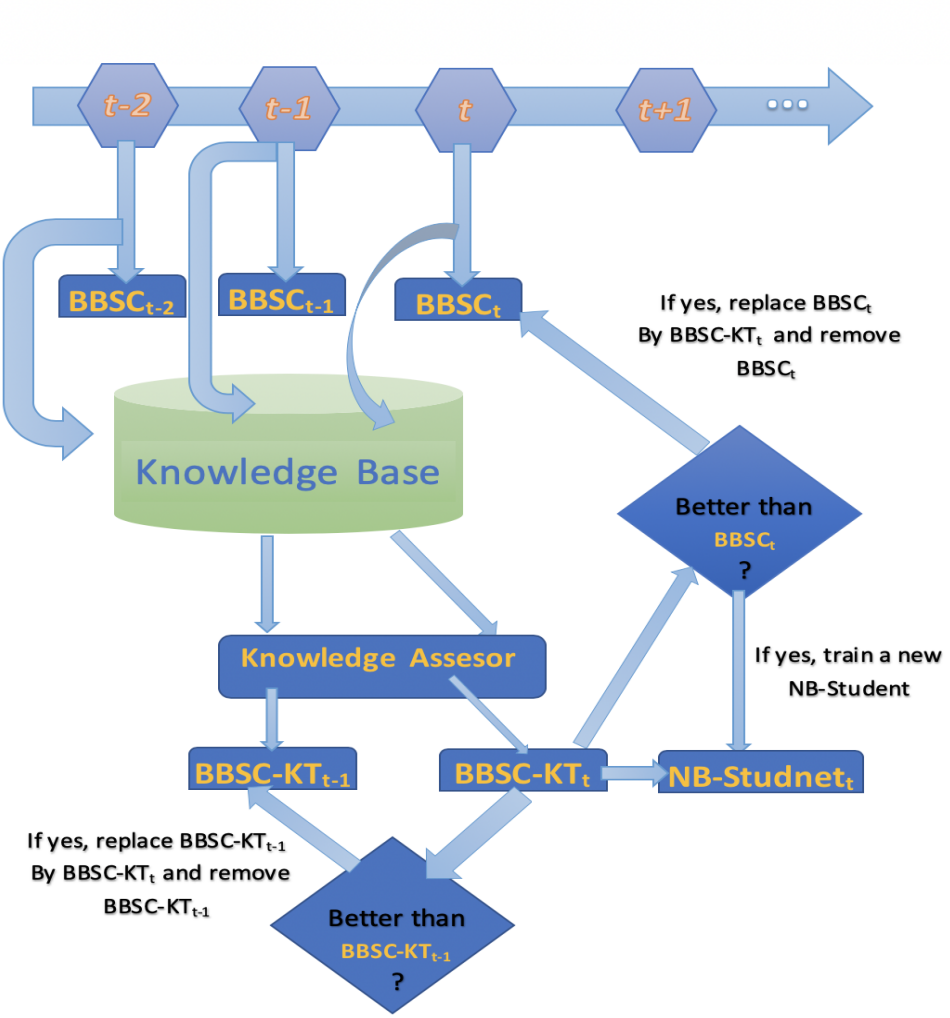

Figure 3. Lifelong Sentiment Classification Architecture.

In order to control the knowledge transfer, the cost function for BBSC (Formula 5) will be replaced by Formula 6 with an extra penalty term.

$$\arg\min_{\theta} \sum_{1}^{N} |p\hat{n}_{+} - p_{n+}| + |p\hat{n}_{-} - p_{n-}| \tag{5}$$

where θ is parameters of BBSC model; p_{n-} is the true probability to a sample to negative(0 or 1); $p\hat{n}_{-}$ is the predicted probability to a sample([0,1]); and the total sample number is N.

$$\arg\min_{\theta} \sum_{1}^{N} |p\hat{n}_{+} - p_{n+}| + |p\hat{n}_{-} - p_{n-}| + \sum_{1}^{K}\sum_{1}^{M} |\hat{p}_{km+} - p_{km+}| + |\hat{p}_{km-} - p_{km-}| \tag{6}$$

where θ is parameters of BBSC-KT model; the total previous domain number is K and the there are M sample in the k-th domain will be choose to join training.

Now, the main issue for BBSC-KT is how to choose the K domains and M samples in each domain. Firstly, there will be a $D_k \in [0, 1]$ to determine whether choosing the k-th domain. $D_k = f_d(s_k)$, where f_d is a function with threshold λ_d and s_k is the task similarity between the k-th domain with the current domain(see Algorithm 1 and section 6.9). As for λ_d, it is depended on the task similarity function. In this paper, the authors use the BBSC model for current domain to test previous k-th domain and

get F1 score as task similarity. Hence, the authors suggest that λ_d should be large than 0.5 because otherwise it means it is worse than random guess.

Due to the imbalance problem is crucial in sentiment classification, it is nature to choose negative review with higher probability when performing knowledge transfer. This paper propose to use $\alpha_- = 1 - N_- / (N_+ + N_-)$ as the probability to choose a negative review. Where N_+ and N_- are total number of positive and negative review.

To further reduce the computation, a sample in previous domain only will be choose when has not be learned well by current domain's model. Which means the current domain's BBSC model fail to predict it. Let use XOR function $\sigma_{km} = \hat{y}_{km} \oplus y_{km}$ to determine it. So the Formula 6 can be represent as:

$$\arg\min_{\theta} \sum_{1}^{N} |\hat{p_{n+}} - p_{n+}| + |\hat{p_{n-}} - p_{n-}| + \sum_{1}^{K} D_k \sum_{1}^{M} \alpha_{km}\sigma_{km} \left(|\hat{p}_{km+} - p_{km+}| + |\hat{p}_{km-} - p_{km-}|\right) \quad (7)$$

When the BBSC-KT model had been trained, it will be used to test upon current domain and previous domains. If it achieved better performance on a domain, the previous "teacher" BBSC or BBSC-KT model will be replaced by current BBSC-KT model. This means that multiple domains will share the share model, which can significantly reduce to the requirement for the storage.

4.7. Scalability Analysis

In order to make the algorithms still could work in large scale big data scenario, the sclability includes storage, training cost and prediction cost needs to be well considered. For sentiment classification, the scalability issue could be very different according to different application scenarios. If only one domain is needed to be predicted once time (offline prediction), then both BBSC or BBSC-KT only need to store one model. In this case, scalability won't limit the usage of the algorithm. However, if a company needs to handle different domains at the same time (online prediction), the situation is much complex. As BBSC is designed for each single domain, so it needs maintain one model for one domain. In application, it needs to load and run all models in the meantime, which requires huge resource. Due to each BBSC model is 1.23 GigaByte and assuming each cloud server's memory size is 16 GigaByte, 1000 domains require at least 72 cloud servers to run the BBSC algorithm. This cost is unacceptable for most small companies. As for BBSC-KT, it improve learning of all tasks symmetrically so only needs load one model for all 20 Amazon domain, so the scalacity is much better than BBSC. But considering a extreme situation that each BBSC-KT model only works upon a single domain, which means the symmetrical characteristic is unavailable, it will face the same problem as BBSC. Details will be provided in the Section 5.

Considering most of sentiment classification tasks don't need real-time response, they could be regards an offline tasks. In this case, it is possible to handle tasks one by one, so each time only needs to load one model. For offline tasks, Lambda Architecture [8] could be used for big data situation. Instead, if a quick response is required for all tasks, the "student model" will be the solution.

4.8. Time Complexity Analysis

This subsection is going to analysis the time complexity for each algorithm mentioned in this paper. Assume there are m tasks and each task has n instances. For Algorithm 1, its time complexity is $O(m * n)$. Due to neural networks always train data with multiple iteration i, so the time complexity of Algorithm 2 and Algorithm 3 are $O(m * i * n)$. Meanwhile, both LSC [12] and LNB [24] are $O(m^2 * i * n)$ at training stage. Then, the Algorithm 4 is $O(m * n \log n)$. In summary, the time complexity of main system is $O(m * i * n)$, it is a polynomial complexity and better than other current works. If only considering the task number m, the time complexity of the whole system is $O(m)$. This indicates that it would not meet a serious scalability issues and also is acceptable in big data scenario.

5. Scalability for Lifelong Learning

As lifelong learning is a continuous learning paradigm, it needs to adapt infinite incoming tasks so its knowledge base also is growing. Hence, the scalability of it could be a limitation during learning. Previous design in this article mainly focus on how to use lifelong learning to improve the learning performance upon tasks, this section will analysis the scalability issue of lifelong learning in big data processing and its role in smart communication, especially for IoT scenario.

5.1. Scalability in Big Data Processing

Scalability in Big Data involves two aspects: improving scalability for current big data processing approaches and improving scalability for new lifelong learning algorithms.

5.1.1. Relation between Lifelong Learning and Big Data Processing

Lifelong learning and big data processing benefit from another symmetrically and and could not work without individually. In order to accumulate knowledge and become more powerful, lifelong learning needs big data as nourishment. Similar, as big data has variety sources and various forms, so it would be hard to integrate the valuable information among "information isolated island" when lacking lifelong learning and knowledge transfer. Lifelong learning could discover most of the value form big data and so it is essential for big data processing.

5.1.2. Improving Scalability for Big Data Processing

When talking about lifelong learning, it is easy to think that it is a much more complex system comparing with traditional algorithms. However, some lifelong designs are raised to improving scalability. For example, ELLA(Efficient Lifelong Learning Algorithm) [22] speeds up the batch multi-task learning over $1000\times$ times. Due to most of big data processing techniques need to learn the representation of data, so this could be a repetitive work and wastes huge times. In lifelong learning, the learning of data representation could be done at first or become an incremental learning. In big data scenario, data value could be huge, so to train a best data representation upon all available data is impossible for most researchers So this should be a fundamental work for big organizations. Princeton University contributed ImageNet [13] for image processing and Google contributed BERT [23] for NLP. Then, any researchers could re-use such fundamental work and to save much time and computing resources. This makes researchers able to solve more tasks with limited resources, which equals to improve the scalability.

5.1.3. Improving Scalability While Promoting Performance

Above subsection had showed that some lifelong learning algorithms could improve the scalability of big data processing. Meanwhile, more lifelong learning is designed to promoting performance by leveraging big data. In this case, the performance is better but the system complexity also is increased due to knowledge accumulation. For example, although NB-T is a non-lifelong algorithm and its performance is poor, its complexity is also low. Instead, NB-ST has better performance, but it's time complexity becomes $O(n^2)$. Similar, BBSC and BBSC-KT needs more computation capacity and data storage. This extra cost could influence the scalability of lifelong learning algorithms and should be kept within reasonable range.

Although the most of lifelong learning algorithms become more complex while promoting performance, it is still possible to improve efficiency and scalability meanwhile. For BBSC, it needs to store a big neural network for each task, it could be big cost. In the same time, BBSC-KT better performance but only needs to keep one model at final. This shows that it is possible to beef up both performance and efficiency by lifelong learning.

5.2. Improving Scalability by Lambda Architecture

Based the scalability analysis for BBSC-KT at Section 4.7, there could be a huge computation cost issue in an extreme case. This would happen only when there are a large number of tasks that need online prediction and one BBSC-KT model is unable to handle all task. Fortunately, this kind of extreme case does not exist in Amazon tasks, due to one BBSC-KT model could solve all tasks and no extra computation resource required. However, in order to avoid scalability problem, such kind of extreme case needs to be considered. The consequence of it is that multiple BBSC-KT models need to be loaded at the same time, but a company might be unable to afford the cost. This difficult situation always exists in big data scenario. The users often want to get the results immediately but computation of big data needs time. A common trade-off is to speed up by decreasing the requirement for the accuracy. Which means using a faster but less accurate algorithms. In the paper, this refers to use the "student model" to provide an online prediction. Although its accuracy is lower than BBSC-KT, the loss is not significant and its response is immediate. This is the online part, which provides a quick and acceptable response to users. As for the offline part, a more accurate result also is available if the users are willing to wait for more time. With more time, it is possible to use a single machine to load multiple BBSC-KT models to solve all tasks. If more computation resources are available, less time needed, it also is a trade off depends on the demands of the user. Combined above online and offline approaches, a more flexible solution could be delivered. In big data processing, it also be could as streaming processing and batch processing [27]. A classical technique for this combined solution is Lambda Architecture (Figure 4), which is common used to enhance the saclability for big data processing.

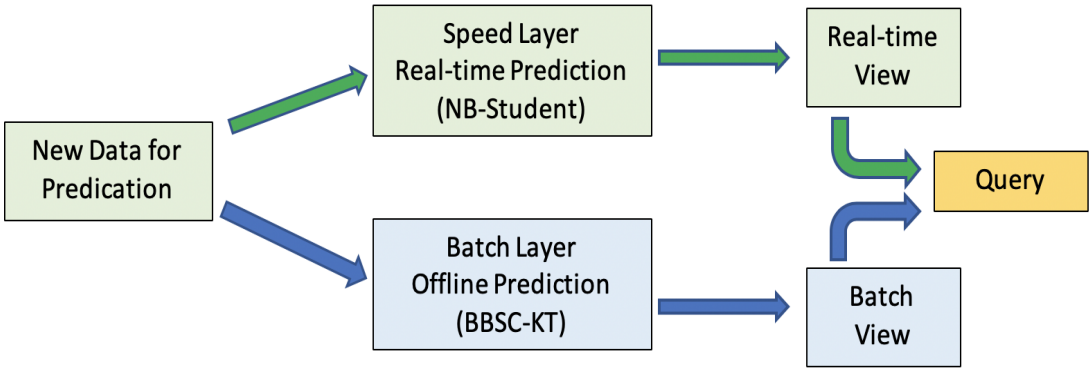

Figure 4. Lambda Architecture for Lifelong Sentiment Classification.

As for the scalability of storage, 10 billion reviews only needs GigaByte. Each BBSC model or BBSC-KT model needs 1.23 GigaByte, even for 1000 domain only needs for 1.23 TeraByte, which is a small cost for any industry scenario. Based on the experiments of this paper, all the 20 Amazon domains could share the same BBSC-KT model, which means the requirement of storage will be much lower that expected in practice.

5.3. Scalability for Smart Communication and Internet of Things

With the explosive growth to internet of things (IoT) and smart phone, smart communication has became as a hot area of big data processing [28]. Comparing with traditional internet, IoT has far more data. In a complex social network, machines generate more data than human [27]. Although IoT could generate plenty of data, the lacking of processing capability causes that it is difficult to discover the value from information. Without data mining, data is just "junk information" rather treasure. As the data volume of IoT has exceeded the capacity of the most big data platforms, edge computing has

turns to a reasonable choice. According to previous design for lifelong learning in the paper, it is a centralized system. Although the main parts of lifelong learning is not suitable for edge computing, some computation could transfer to smart device. For example, BBSC-KT is too large for IoT device, but NB-Student is possible to install on users' smart device to reduce the cost of cloud server (Figure 5). From this angle, IoT and social network has increased the data processing challenge for lifelong learning, but also provide an opportunity. In order to make sure the scalability of lifelong learning, the architecture should implant more components to smart device. In addition, although this approach could reduce the cost for cloud server, the cost for communication among users would significant increase. Hence, this approach should ask the agreement from users and ensure the privacy safety [29].

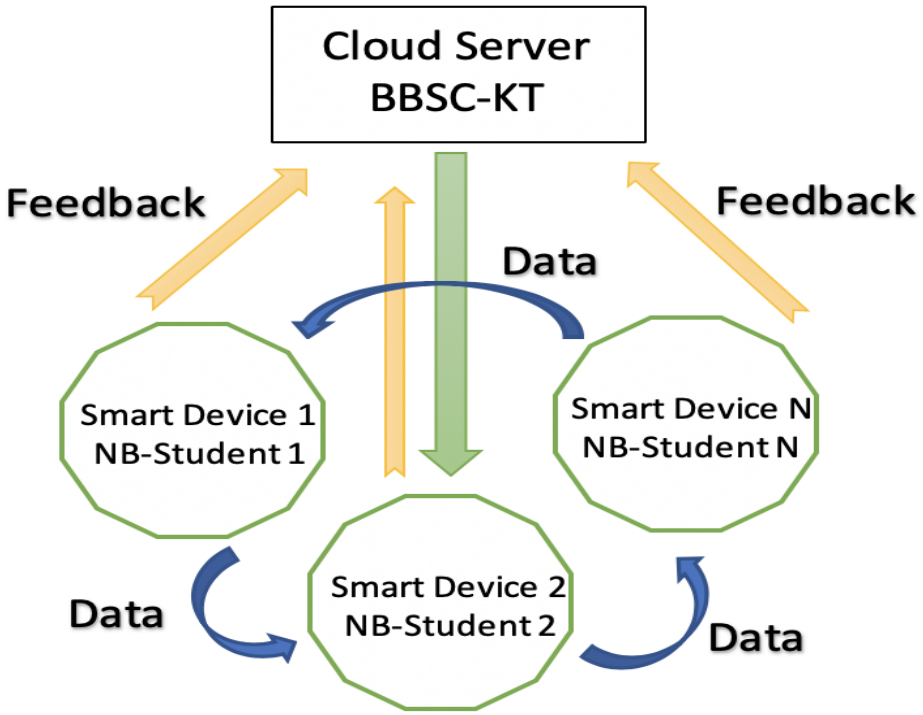

Figure 5. Edge Computation for Lifelong Sentiment Classification [23].

6. Experiment and Results

6.1. Datasets

As this paper uses LSC [12] as a baseline, its datasets also are used for this experiment. It contains 20 domains' Amazon customer review and 1000 reviews for each domain. As its data distribution is imbalanced, this paper will mainly focus on the minor class.Following parts will figure out how implicit knowledge and explicit knowledge influence algorithm's performance upon target task. Naïve Bayesian and BERT are chose to discussed.As this dataset is imbalanced, it is much difficult for an algorithm to get high performance on minority class(negative reviews) than the major class. Hence, this paper will mainly focus on the comparison upon negative class.

IMDB dataset also is used as a supplement. It is movie review dataset and is quit different with the Amazon dataset. It will help the researchers to figure out how task similarity influences the lifelong leanring.

6.2. Experiment Environment

The experiment is conducted upon Google Colab and Google Cloud AI Platform, the algorithm running time depends on Google's equipment allocation.

6.3. Data Pre-Processing

In order to let Naïve Bayesian model and BBSC model could understand the text, the text needs to be converted to token at first, which is tokenization. In addition, the BBSC model only could receive sentence embedding as input, so an embedding procedure will be conducted.

6.3.1. Tokenization

Each sentence is composed by words or tokens, most of NLP models taken token as the basic input unit. For example, a sentence "This here's an example of using the tokenizer", which would be converted to a series of token: ['this','here',"'",'s','an','example','of','using','the','token','##izer'].

6.3.2. Embedding

For BBSC model, embedding includes token embedding and sentence embedding. Token embedding is the vocabulary IDs for each of the tokens. There are three special token:

1. CLS: A token to indicate the start of a sequence.
2. SEP: A token to separate each sequence.
3. MASK: Token used for mask technique, to help the training of language model

Sentence Embedding (Figures 6 and 7) aims to distinguish two sentences (There only one sentence is needed in sentiment classification).

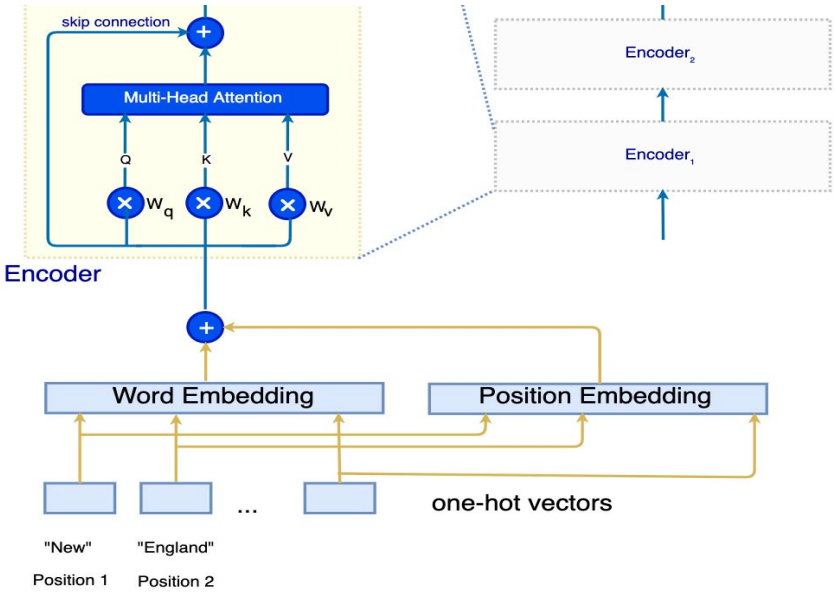

Figure 6. BERT Model in Details [23].

```
INFO:tensorflow:tokens: [CLS] what a great little alarm clock . it
INFO:tensorflow:tokens: [CLS] what a great little alarm clock . it
INFO:tensorflow:input_ids: 101 2054 1037 2307 2210 8598 5119 1012
INFO:tensorflow:input_ids: 101 2054 1037 2307 2210 8598 5119 1012
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Figure 7. Embedding Outcome Example.

6.4. Naïve Bayesian Classifier

Naïve Bayesian approach stores the knowledge in the probability distribution, which shows that how the model estimate the sentiment of each word. As there are 20 domains, so we evaluated the performance of transfer learning over all domains.

We implemented three algorithms:

- NB-T: Naïve Bayesian approach only trained on target task
- NB-S: Naïve Bayesian approach only trained on source tasks
- NB-ST: Naïve Bayesian approach trained on both source and target tasks

Here, source domains refer to all domains except the target domain. And the Macro-F1 was chose as it is average f1 score upon both classes and shows how an algorithm performs on minor class when data is imbalanced. All the text had removed the stopped words by NLTK and each domain was tested by 5 fold cross validation.

As the experiment results show in Figures 8 and 9, NB-S performs as good as NB-T. It's a little surprised because that the NB-S didn't received any information from the target domain. This shows that power and potential of knowledge transferring. It's also worth to noticed that NB-ST is much better than the NB-T. This means that the target domain indeed benefits from the knowledge of source domains. The authors believe that there is a global probability distribution of the sentiment of all words. The probability model trained on a single domain only saw a few of words, so the probability estimation is not complete enough. So with more knowledge from source domains, the performance becomes better. However, we also should noticed that, the performances of NB-ST upon two domains (Figure 8): Gloves and MoviesTV is lower. This warns that the inappropriate knowledge transferring will decrease the performance.

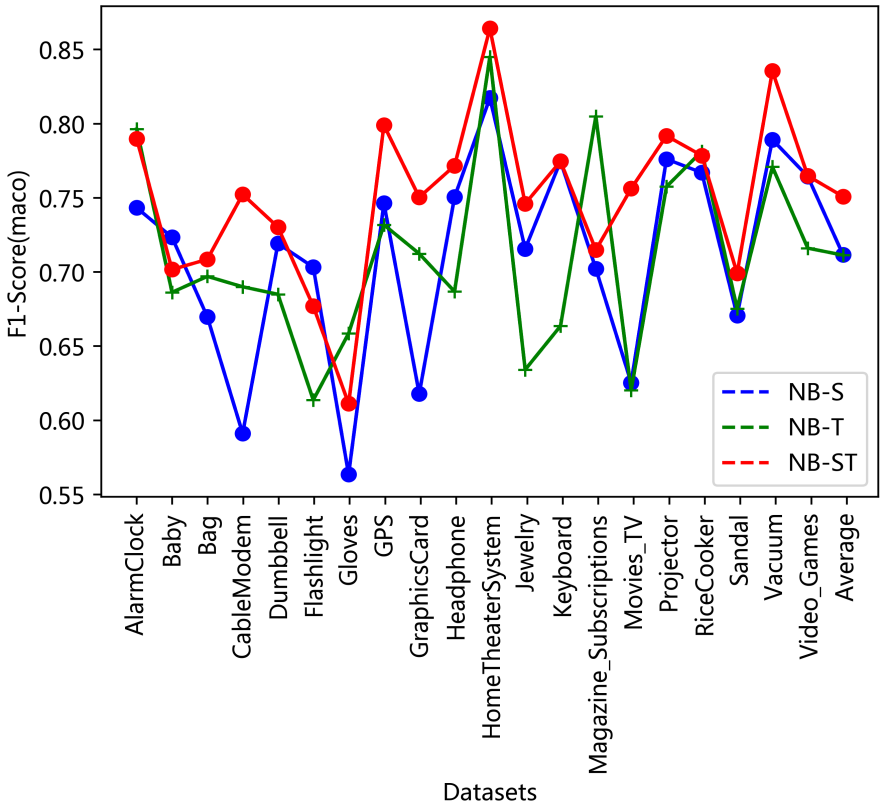

Figure 8. F1 Scores over 20 Domains, NB-ST comparing with NB-S, NB-T.

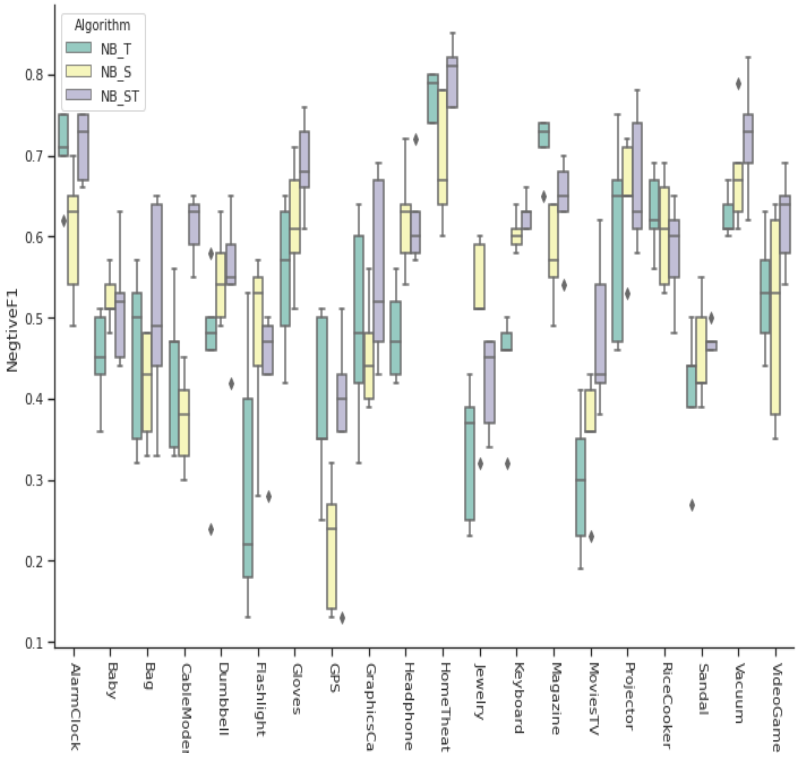

Figure 9. F1 Scores upon Negative Class over 20 Domains, NB-ST comparing with NB-S, NB-T.

6.5. BERT Based Model for Sentiment Classification

The BBSC model is fine-tuned based on the pre-trained model released by "TensorFlow Hub" and which is the "base" version of BERT. From Figure 10, it's obvious that BSCC is much better than NB-ST. BBSC has higher F1 score over negative class upon almost all domains except CableModem. Hence, it is reasonable to use BSCC model to teach a Naïve Bayesian model.

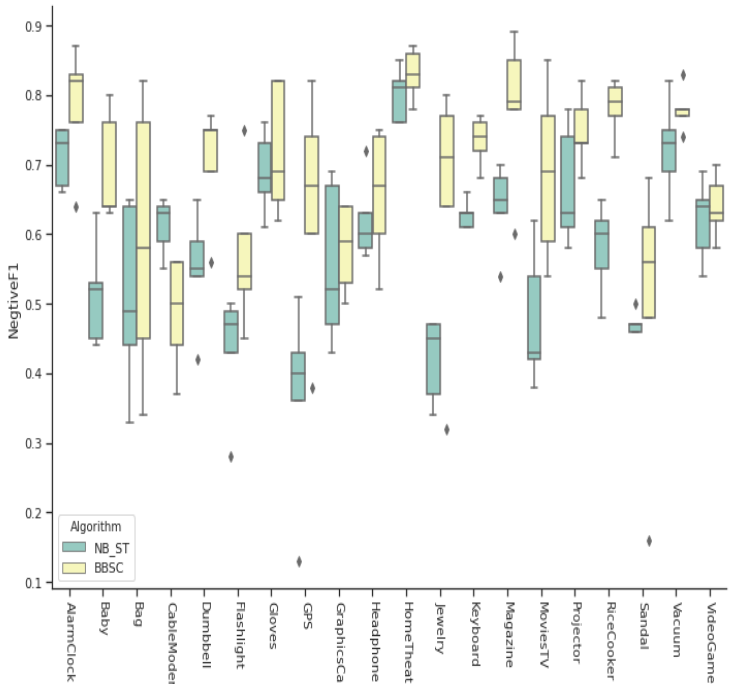

Figure 10. F1 Scores upon Negative Class over 20 Domains, BBSC comparing with NB-ST.

According to the Table 1, we can see that the student model 'NB-Student' which is trained with pseudo labels provided by BSSC is much better than NB-ST. Due to the NB-Student is trained by BSSC, the performance of it is close on BSSC rather than NB-ST. This due to when the labelled data is sufficient, the performance of models will mainly dependent on the data quantity rather the model design.

6.6. BBSC with Knowledge Transfer

Figure 11 is the comparing between BBSC-KT and BBSC upon negative class. Superised, BBSC-KT performs better than BBSC upon all domain after the whole training process. As BBSC-KT has the same model size with BBSC, so it is reasonable to remove previous BBSC models from knowledge base. This result leads to a good feature and inspire the authors that although BBSC is designed for a single domain, the knowledge transfer could help it to also work well on a group of tasks. Hence, the capacity requirement for lifelong learning architecture might be significantly lower than expected as it could transfer knowledge symmetrically.

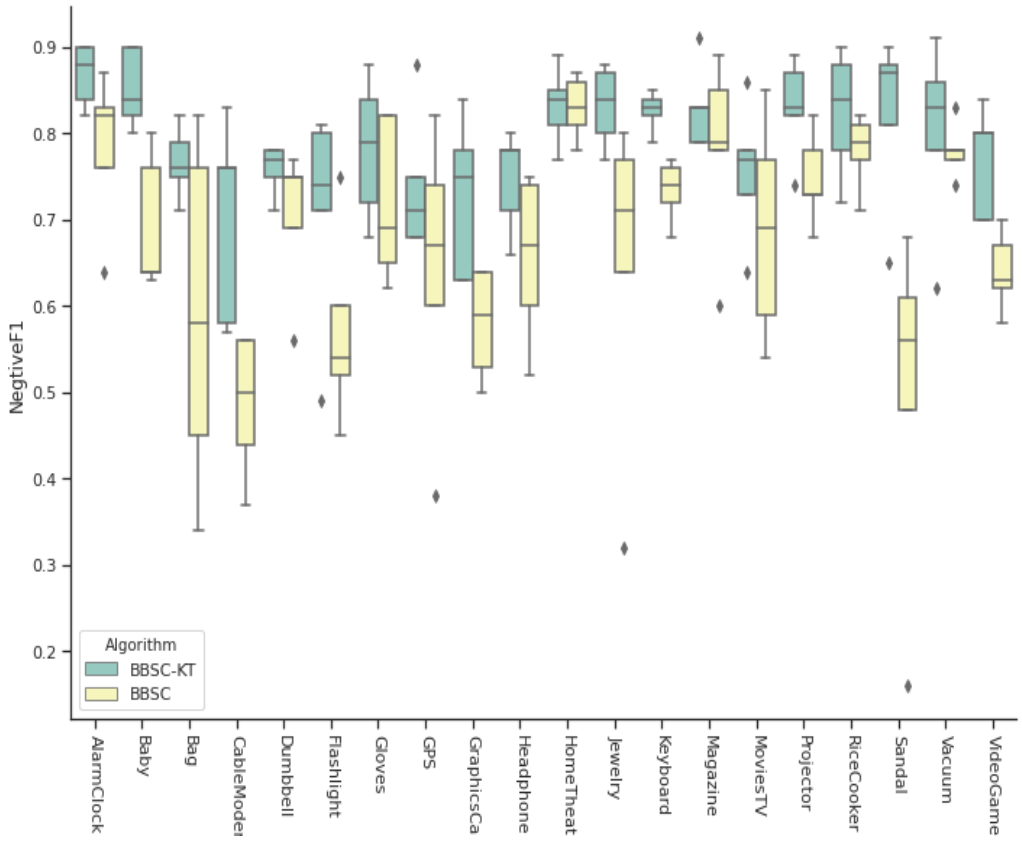

Figure 11. F1 Scores upon Negative Class over 20 Domains, BBSC-KT comparing with BBSC.

From Table 1, it is clear that even BBSC is significantly better than LSC and sightly advanced to LNB. Noticed that, LSC has knowledge transfer and LNB has bidirectional and symmetric transfer while BBSC only obtained knowledge during pre-training periods of BERT. BBET could use for other except sentiment classification, but LSC and LNB not. Remarkable, BBSC-KT obtained a large promotion by controlled knowledge transfer. With the large corpus that BERT learned, it is easy to take advantage when comparing LSC and LNB.

Table 1. F1-score for the negative classes in the Natural Distribution.

Average F1 Scores Comparing						
NB-T	NB-S	NB-ST	LSC [12]	LNB [24]	BBSC	BBSC-KT
45.20	55.00	56.49	56.62	64.96	67.78	**78.84**

6.7. Consuming and Efficiency Comparing

Although Figure 10 has showed that BBSC's performance is better than NB-ST, we still need to test whether NB-Student can learn well from it. According to Table 2, NB-Student's F1 socre is 0.9233, 97.44% comparing with BBSC. Though the performance's is sightly lower than BBSC, it is still an acceptable performance for the most of industry scenarios.

In real industry environment, the inference time consuming also is the main concern. According to the Table 3, BBSC's train time cost is acceptable, but the inference speed is big problem. Nobody could wait 8 s for a single prediction, so the BBSC is definitely unsuitable for deploying in production environment. Comparing, NB-Student speeds around 2422 times, which is a reasonable speed.

Table 2. Resource Consuming Comparing.

Model	Parameter Number	Model Storage	F1 Score
BBSC	Around 110 Millions	1230 MB	**0.9476**
NB-ST	0.012 Millions	0.257 MB	0.7509
NB-Student	**0.004 Millions**	**0.096 MB**	**0.9233**
NB-Student/BBSC	1/27500	1/12183	97.44%

Table 3. Time Complexity Comparing.

Model	Train Time Cost	Test Time Cost (Single Instance)	GPU Required
BBSC	126.567590 s	8.657926 s	Yes
NB-Student	**7.944083 s**	**0.003574 s**	No
NB-Student/BBSC	1/16	1/2422	—

6.8. Scalability in Big Data Scenario

This section will analysis how Lambda architecture improve the scalability of lifelong learning and provides a more flexible solution for industry. This experiment assumes that each task contains 1000 reviews equally and each cloud server has 16 GigaByte. Due to the neural network predicts instance a batch per time, so its time usage for per prediction instance is different and depends on the batch size. So the time for 1000 instances is not equal to 1000 times more than the time for single instance.

According to Table 4 and Figure 12, both BBSC-KT and NB-Student could conduct 1000 tasks within 14 h. However, if less time required, the cost for computation would significantly increase. As BBSC-KT needs GPU server which is much expensive, so it is not suitable for real-time processing. So, the authors suggest that BBSC-KT only is required to be trained offline regularly (e.g., 10 GPU servers could handle 10,000 tasks within two week). NB-Studnet is more suitable for real-time processing.

Table 4. Time Complexity Comparing for One Task (1000 test instances).

Model	Train Time Cost	Test Time Cost (1000 Instances)	Total Cost	GPU Required
BBSC-KT	126.567590 s	617.653231 s	744.220821	Yes
NB-Student	**7.944083 s**	**4.137653 s**	**12.081736 s**	No

Due to the training is not required as real-time, so the scalability issue is mainly associated with the prediction. Figure 13 shows how much servers required for real-time prediction for specific task number. It is easy to see, only 7 CPU servers could handle 10,000 tasks (1000 reviews per task) in real-time. Noticed that, 10,000 tasks have 100 million reviews and this closes to the maximum review number for a single company. This kind of processing capacity is enough even for Amazon or Taobao.

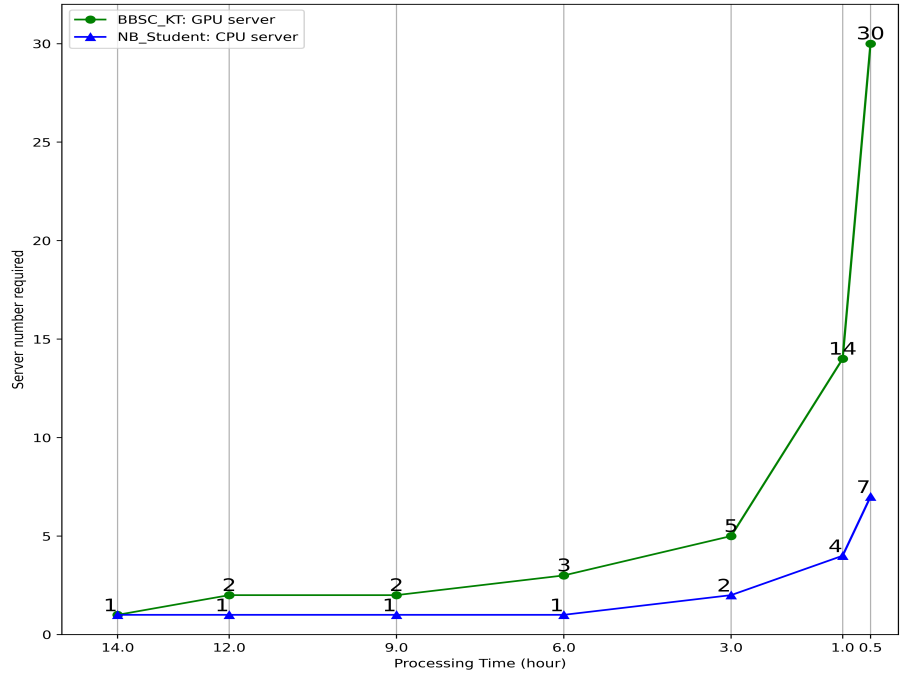

Figure 12. Number of Server required for 1000 Tasks within specific Time.

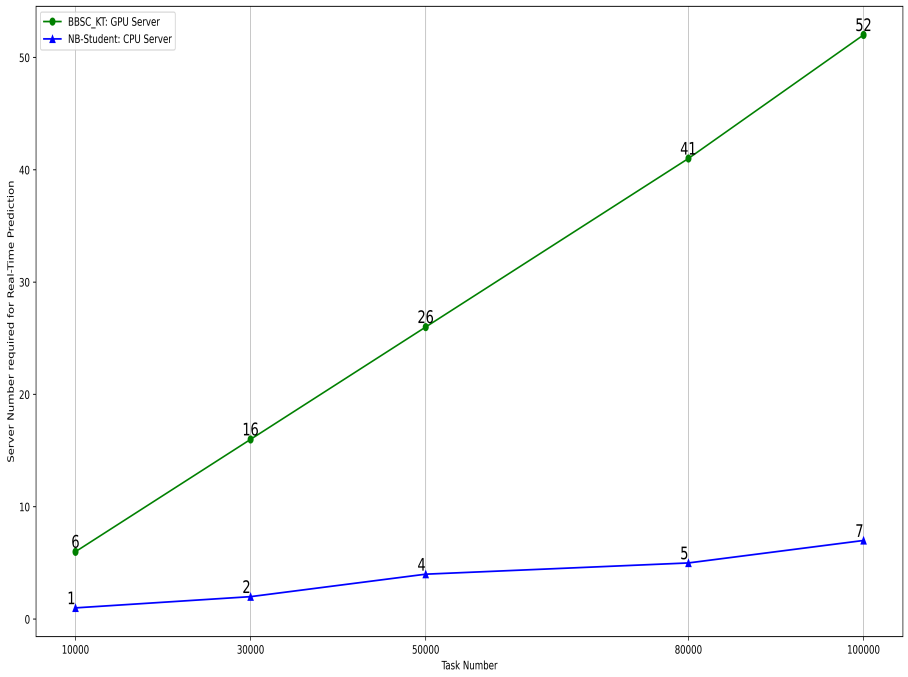

Figure 13. Number of Server required for Real-Time Processing.

In summary, combined with offline training of BBSC-KT and NB-Student and real-time prediction of NB-Student, the lifelong learning system could handle a large scale sentiment classification with high performance. In order to increase the scalability and reduce the cost, Lambda architecture (Figure 14) is suggested. In this system, the user could determine whether to use NB-student or BBSC-KT for each task based on the performance requirement and budget.

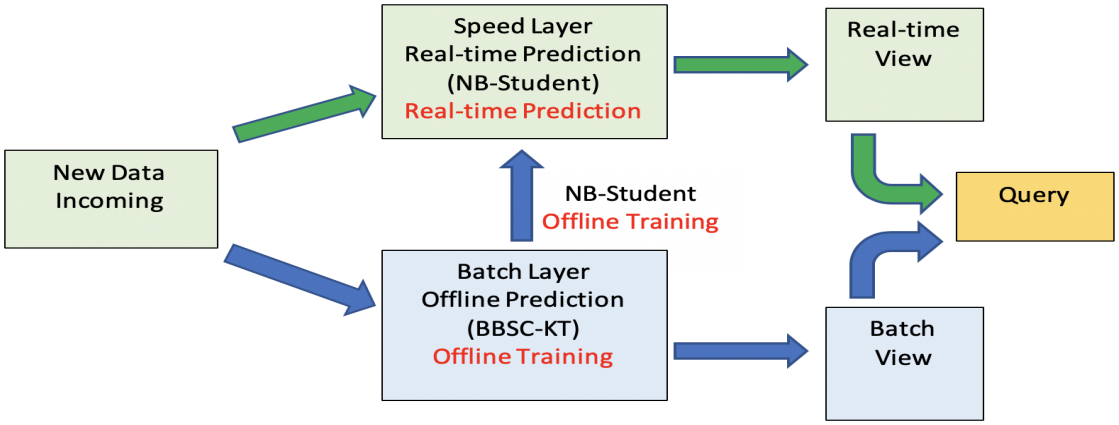

Figure 14. Lambda Architecture for Lifelong Sentiment Classification.

6.9. Novel Task Detection

In the experiment, there had been 21 task domains. 20 of them are Amazon product reviews and the last one is the IMDB movie reviews. All of them had been introduced in Section 5.1. Expect using unsupervised method to calculate similarity, we also could use previous classifier to validate the new task. Here we collected the BSCC classifiers for the Amazon tasks and used them to predict the rest tasks. According to the Figure 15, Amazon tasks' classifiers could achieve a good performance upon the rest Amazon tasks while all they perform worst upon the IMDB task. This leads an obvious conclusion: the IMDB task is much different with previous Amazon tasks and there are any classifier could handle it, so it is a novel task. So, the 'supervisor' starts to train a new BSCC model and another student model.

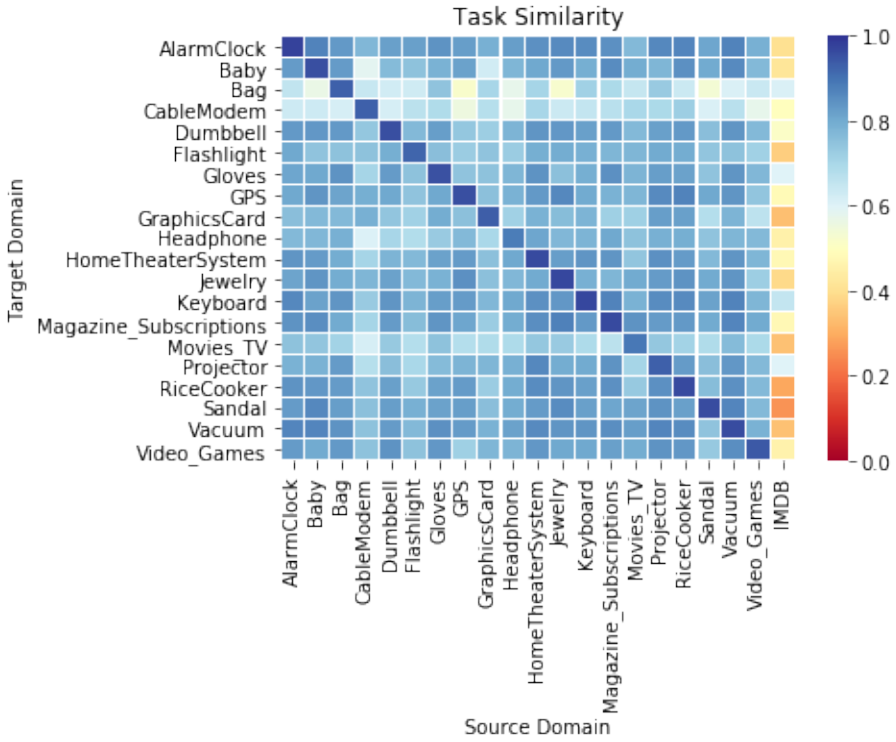

Figure 15. Task Similarity Analysis. Value in Heatmap refers to the F1 score of prediction. Classifiers belong to the source domain tasks and target domain data is used to predict.

6.10. Active Learning

In order to test the active learning approach, we assume the IMDB dataset as a novel task to test. As IMDB has 50 K movie reviews, we only sampled part of them because it is too hard to afford the computation. For test data, we keep the amount as 5000 instances. As the train Data, we firstly choose 5 K and then decrease to 0.5 K. Table 5 shows wehn train data amount is 5 K, the BBSC model with activate learning only improved the performance with 0.19%. Turn to the case of 0.5 K train data, the BBSC model has been improved by 4.4%, which is a significant improvement.

Table 5. Active Learning.

Model	Train Data Amount	Train Data Choose Method	F1 Score
BBSC	5k	Choose Randomly	0.8660
*BBSC**	5k	Choose by Active Learning	**0.8679**
BBSC	0.5k	Choose Randomly	0.7719
*BBSC**	0.5k	Choose by Active Learning	**0.8159**

7. Discussion

This research had combined most of previous achievements for lifelong machine learning, demonstrated them as an integrated case. The novel task detection is inspired by the online learning (or called incremental learning [30–32]) without task segment [15]. Involving this because lifelong learning always faces to novel tasks and there aren't a hard boundary among tasks. The training process of BBSC model is in fact a typical case of transfer learning and domain adaptation [33] with implicit knowledge transfer. BBSC-KT further improves BBSC by symmetric transfer learning. "Student Model" leverage the principle of knowledge distilling [19] and model compression [34,35]. When the system is suffering from lacking of data, it is nature to ask for more data. Hence, the choice for data annotation also is essential, which called active learning [21,36] for an intelligent algorithm. The authors had tried to use a single technique like multi-task learning [37] or domain adaptation, none of them succeed. Such failures leads the authors for reflection and finally realized that only an high-level architecture with the integration of those techniques could lead us to the lifelong learning.

Noticed that, there are two ways to implement lifelong learning for classification. A way is only to use manually defined explicit knowledge like LSC [12] and LBN [24] and another way is to show implicit knowledge among all tasks like BERT and this paper. This approach could widely share knowledge among tasks without limits of task requirement due to all NLP tasks could share the same language model. This paper also proved that this way is more effective.

Although this research had leveraged the transfer learning to deal with the novel task, there still are space for improvement. For example, when training IMDB task, the previous classifiers for Amazon task had been supposed to be useful due to their tasks are similar. However, the Amazon tasks' classifiers seem useless for IMDB task. This may be caused by lacking of a good knowledge validation and domain adaptation. The author believe more meta-learning [38–40] should be involved to improve. With more meta-feature [41,42], task relationship could be more clear and the usage of meta-knowledge [43] would be more efficient.

Considering this paper only used Naïve Bayesian classifier to train the "student model", there might are other classifiers have better performance and more suitable. Small scale neural networks could be an option. Had involved ensemble learning approaches like the Xgboost [44], the student model even had potential to learn beyond the "trainer model".

An important but lacking of discussion area for lifelong learning is the scalability for big data processing. Social network, smart communication bring more data and challenges but also provide more opportunities.

During the active learning, due to plenty of clustering methods and distance calculation formulas are available, there still is chances to enhance the active choosing algorithm. The authors believe that the active learning part is a most potential research direction for lifelong learning. At current stage of industry development, collecting more labelled data still is the first choice when meeting problem. At this time, active learning would become as the most urgent research demand.

Finally, one part that is easiest to be ignored in machine learning architecture is the security part. Before deploying it to the production environment, necessary testing [45,46] should be included.

8. Materials and Methods

This research involved two datasets, Amazon reviews could be found by previous paper [5] and the IMDB reviews is open source. Consider IMDB is a large dataset, we suggest to sample them, otherwise might cause out-of-memory error.

The BERT model mentioned above is the BERT-Base, which also is open source. Computation capacity might limit researchers follow this research, hence cloud server is recommended, most AI platform is enough to continue this research.

9. Conclusions

This paper proposed a multi-layer lifelong learning architecture and provided a novel symmetric BBCS-KT model for sentiment classification as a demonstration. As this design doesn't limit to any specific classifier, it could be migrated to any classification scenario.

The authors argued that the core of lifelong learning is to adapt to the novel task. Hence, novel task detection is necessary and transfer learning by pre-trained model could help to adapt to the novel task. Knowledge Assessor was proposed as a key component to help the control of knowledge transfer to a novel task. To deal with the efficiency problem of large scale of pre-trained model, this paper additional proposed a knowledge distilling approach to speed up the inference part of our architecture. In order to keep high scalability in big data scenario, the author raised to use Lambda architecture to balance the demands for high performance and quick response. Edge computation also was discussed for smart communication and suggested to deal with the information explode. Active learning also was mentioned in this paper to solving the problem of lacking of labelled data, which could help to improve performance of model with limited resource. The most important point needs to be indicated is that the storage of lifelong system could continuously increase, so it is crucial to consider the efficiency. And when the knowledge amount gradually beyond storage capacity, knowledge forgetting problem is need to pay more attention.

As for the further research, we will continue for and suggest researcher community to focus on the knowledge transfer validation and active learning. These techniques would have tend as a decisive role when lacking of data.

Author Contributions: Conceptualization, S.-U.G. and X.H.; methodology, S.-U.G. and X.H.; software, X.H.; validation, X.H.; formal analysis, X.H.; investigation, S.-U.G. and X.H.; resources, X.H.; data curation, X.H.; writing–original draft preparation, X.H.; writing–review and editing, X.H.; visualization, X.H.; supervision, S.-U.G., K.M. and P.W.; project administration, S.-U.G., K.M. and P.W.; funding acquisition, S.-U.G. and K.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the CERNET Innovation Project under Grant NGII20161010.

Acknowledgments: This research is funded by Research Institute of Big Data Analytics and Department of Computer Science and Software Engineering, Xi'an Jiaotong-Liverpool University. Thanks Hao Wu (undergraduate student at University of Liverpool) for providing partial experiment results for Naïve Bayesian.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LML	Lifelong Machine Learning
NB	Naïve Bayesian
BERT	Bidirectional Encoder Representations from Transformers
KB	knowledge base
BBSC	BERT based Sentiment Classification

References

1. Thrun, S.; Mitchell, T.M. Lifelong robot learning. *Robot. Auton. Syst.* **1995**, *15*, 25–46. [CrossRef]
2. Thrun, S. Is learning the n-th thing any easier than learning the first? In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 1996 ; pp. 640–646.
3. Thrun, S. Lifelong learning algorithms. In *Learning to Learn*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 181–209.
4. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
5. Chen, Z.; Liu, B. Lifelong machine learning. *Synth. Lect. Artif. Intell. Mach. Learn.* **2016**, *10*, 1–145. [CrossRef]
6. Pal, G.; Li, G.; Atkinson, K. Big data ingestion and lifelong learning architecture. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 10–13 December 2018; pp. 5420–5423.
7. Nallaperuma, D.; Nawaratne, R.; Bandaragoda, T.; Adikari, A.; Nguyen, S.; Kempitiya, T.; De Silva, D.; Alahakoon, D.; Pothuhera, D. Online incremental machine learning platform for big data-driven smart traffic management. *IEEE Trans. Intell. Transp. Syst.* **2019**, *20*, 4679–4690. [CrossRef]
8. Pal, G.; Hong, X.; Wang, Z.; Wu, H.; Li, G.; Atkinson, K. Lifelong Machine Learning and root cause analysis for large-scale cancer patient data. *J. Big Data* **2019**, *6*, 1–29. [CrossRef]
9. Cui, X.; Bollegala, D. Self-Adaptation for Unsupervised Domain Adaptation. In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019), Varna, Bulgaria, 2–4 September 2019; pp. 213–222.
10. Cui, X.; Al-Bazzas, N.; Bollegala, D.; Coenen, F. A Comparative Study of Pivot Selection Strategies for Unsupervised Domain Adaptation. *Knowl. Eng. Rev.* **2018**. [CrossRef]
11. Mitchell, T.; Cohen, W.; Hruschka, E.; Talukdar, P.; Yang, B.; Betteridge, J.; Carlson, A.; Dalvi, B.; Gardner, M.; Kisiel, B.; et al. Never-ending learning. *Commun. ACM* **2018**, *61*, 103–115. [CrossRef]
12. Chen, Z.; Ma, N.; Liu, B. Lifelong learning for sentiment classification. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Beijing, China, 26-31 July, 2015; Volume 2, pp. 750–756.
13. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
14. Riemer, M.; Klinger, T.; Bouneffouf, D.; Franceschini, M. Scalable recollections for continual lifelong learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Hawaii, HI, USA, 29 January 2019; Volume 33, pp. 1352–1359.
15. Harrison, J.; Sharma, A.; Finn, C.; Pavone, M. Continuous Meta-Learning without Tasks. *arXiv* **2019**, arXiv:1912.08866 .
16. Vinyals, O.; Blundell, C.; Lillicrap, T.; Kavukcuoglu, K.; Wierstra, D. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*; Proceedings.com: Red Hook, NY, USA, 2016 ; pp. 3630–3638.
17. Cover, T.; Hart, P. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory* **1967**, *13*, 21–27. [CrossRef]
18. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
19. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531.

20. Nguyen, T.C.; Pham, T.N.; Nguyen, M.C.; Nguyen, T.T.; Ha, Q.T. A Lifelong Sentiment Classification Framework Based on a Close Domain Lifelong Topic Modeling Method. In Proceedings of the Asian Conference on Intelligent Information and Database Systems, Phuket, Thailand, 23-26 March 2020; pp. 575–585.
21. Settles, B. Active Learning Literature Survey. *Science* **1995**, *10*, 237–304.
22. Ruvolo, P.; Eaton, E. ELLA: An efficient lifelong learning algorithm. In Proceedings of the International Conference on Machine Learning, Atlanta, GE, USA, 16–21 June 2013, pp. 507–515.
23. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
24. Wang, H.; Liu, B.; Wang, S.; Ma, N.; Yang, Y. Forward and Backward Knowledge Transfer for Sentiment Classification. *arXiv* **2019**, arXiv:1906.03506.
25. Hong, X.; Wong, P.; Liu, D.; Guan, S.U.; Man, K.L.; Huang, X. Lifelong Machine Learning: Outlook and Direction. In Proceedings of the 2nd International Conference on Big Data Research, Weihai, China, 27–29 October 2018; pp. 76–79.
26. McCallum, A.; Nigam, K. Text classification by bootstrapping with keywords, EM and shrinkage. *Unsupervised Learning in Natural Language Processing* 21 June 1999. Available online: http://www.ai.sri.com/~kehler/unsup-acl-99.html (accessed on 20 April 2020)
27. Thai, M.T.; Wu, W.; Xiong, H. *Big Data in Complex and Social Networks*; CRC Press: Boca Raton, FL, USA, 2016.
28. Huang, X.L.; Ma, X.; Hu, F. Editorial: Machine Learning and Intelligent Communications. *Mob. Netw. Appl.* **2018**, *23*, 68–70. [CrossRef]
29. Bo, W.; Zhang, Y.; Hong, X.; Sun, H.; Huang, X. Usable Security Mechanisms in Smart Building. In Proceedings of the 2014 IEEE 17th International Conference on Computational Science and Engineering, Chengdu, China, 19–21 December 2014; pp. 748–753.
30. Gepperth, A.; Hammer, B. Incremental learning algorithms and applications. In Proceedings of the ESANN 2016 , European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges, Belgium, 27–29 April 2016.
31. Zhang, J.; Hong, X.; Guan, S.U.; Zhao, X.; Xin, H.; Xue, N. Maximum Gaussian mixture model for classification. In Proceedings of the 2016 8th International Conference on Information Technology in Medicine and Education (ITME), Fuzhou, China, 23–25 December 2016; pp. 587–591.
32. Hong, X.; Zhang, J.; Guan, S.U.; Yao, D.; Xue, N.; Zhao, X.; Huang, X. Incremental Maximum Gaussian Mixture Partition for Classification. In Proceedings of the 2017 2nd Joint International Information Technology, Mechanical and Electronic Engineering Conference (JIMEC 2017), Chongqing, China, 4–5 October 2017; Atlantis Press: Paris, France, 2017.
33. Fang, F.; Dutta, K.; Datta, A. Domain adaptation for sentiment classification in light of multiple sources. *INFORMS J. Comput.* **2014**, *26*, 586–598. [CrossRef]
34. Buciluǎ, C.; Caruana, R.; Niculescu-Mizil, A. Model compression. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, 20–23 August 2006; pp. 535–541.
35. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
36. Sun, G.; Cong, Y.; Xu, X. Active Lifelong Learning With "Watchdog". In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
37. Zhang, Y.; Yang, Q. A survey on multi-task learning. *arXiv* **2017**, arXiv:1707.08114.
38. Afolabi, D.; Guan, S.U.; Man, K.L.; Wong, P.W.; Zhao, X. Hierarchical meta-learning in time series forecasting for improved interference-less machine learning. *Symmetry* **2017**, *9*, 283. [CrossRef]
39. Zhao, X.; Guan, S.S.U. A subspace recursive and selective feature transformation method for classification tasks. *Big Data Anal.* **2017**, *2*, 10. [CrossRef]
40. Zhou, Y.; Bollegala, D. Unsupervised Evaluation of Human Translation Quality. In Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management(IC3K 2019), Porto, Portugal, 9–11 November 2019.

41. Chen, Y.; Dong, H.; Wang, W. Topic-graph based recommendation on social tagging systems: A study on research gate. In Proceedings of the 2018 International Conference on Data Science and Information Technology, Madrid, Spain, 1–2 October 2018; pp. 138–143.
42. Dong, H.; Wang, W.; Coenen, F. Learning Relations from Social Tagging Data. In Proceedings of the Pacific Rim International Conference on Artificial Intelligence, Nanjing, China, 28–31 August 2018; pp. 29–41.
43. Lee, J.; Oh, S.; Dong, H.; Wang, F.; Burnett, G. Motivations for self-archiving on an academic social networking site: A study on researchgate. *J. Assoc. Inf. Sci. Technol.* **2019**, *70*, 563–574. [CrossRef]
44. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
45. Huang, W.; Sun, Y.; Huang, X.; Sharp, J. testRNN: Coverage-guided Testing on Recurrent Neural Networks. *arXiv* **2019** arXiv:1906.08557.
46. Huang, W.; Sun, Y.; Sharp, J.; Huang, X. Test Metrics for Recurrent Neural Networks. *arXiv* **2019** arXiv:1911.01952.