Transductive Transfer Learning in Genetic Programming for Document Classification

Wenlong Fu, Bing Xue(✉), Mengjie Zhang, and Xiaoying Gao

School of Engineering and Computer Science, Victoria University of Wellington, PO Box 600, Wellington, New Zealand
wenlong.fu@msor.vuw.ac.nz,
{bing.xue,mengjie.zhang,xiaoying.gao}@ecs.vuw.ac.nz

Abstract. Document classification tasks generally have sparse and high dimensional features. It is important to effectively extract features. In document classification tasks, there are some similarities existing in different categories or different datasets. It is possible that one document classification task does not have labelled training data. In order to obtain effective classifiers on this specific task, this paper proposes a Genetic Programming (GP) system using transductive transfer learning. The proposed GP system automatically extracts features from different source domains, and these GP extracted features are combined to form new classifiers being directly applied to a target domain. From experimental results, the proposed transductive transfer learning GP system can evolve features from source domains to effectively apply to target domains which are similar to the source domains.

Keywords: Genetic programming · Document classification · Transfer learning · Text classification

1 Introduction

Document classification, as a task of natural language processing, has been addressed by many artificial intelligence algorithms [1,11,14,19]. Since many different words are used in documents, document classification tasks generally have high dimensional and sparse features. In a document classification task, pre-processing, feature construction, feature selection, and model building are normally included. How to select and extract features is very important in document classification [1]. One popular approach is to use the frequency of words to construct a high dimensional and sparse set of basic features [14]. However, there is no sufficient work of how to effectively and automatically extract the frequencies of words to combine as high-level features [6,22].

Genetic programming (GP) has been successfully employed to select features for different tasks, such as edge detection [7], symbolic regression problems [4,5] and a question-answer ranking task [2]. In a web document ranking task [2], GP can effectively select a smaller set of features based on the domain knowledge

Y. Shi et al. (Eds.): SEAL 2017, LNCS 10593, pp. 556–568, 2017.
https://doi.org/10.1007/978-3-319-68759-9_45

than other feature selection methods. Also, GP has been used to evolve programs of combining existing features for document classification [6,22]. It shows that GP has potential to automatically extract features in text classification to construct effective classifiers.

In document classification tasks, we might have pre-defined labels for some datasets, but no labels for other datasets. If these datasets have similar targets and the categories in these datasets are relatively similar to each other, the features extracted from the datasets with pre-defined labels might be helpful for those datasets without labels. Transfer learning has been investigated in this field [18,20]. Transfer learning techniques [18,20] have been used to train models from one source domain, then transfer learnt knowledge to other similar domains [17]. When a transductive transfer learning technique is used to build classifiers, pre-defined category labels are considered in source domains, but category labels are not considered in target domains. It is promising to employ GP to automatically extract features from source domains (including predefined category labels) to apply to a target domain (without considering category labels).

1.1 Goals

The overall goal of this paper is to initially investigate transductive transfer learning in GP for automatically extracting features for document classification. GP is used to automatically extract features from source domains when category labels exist in these source domains, then the extracted features are directly applied to a target domain without training. Features extracted by GP from these source domains are combined to classify documents from target domains. Here, we will investigate the performance on single or multiple features extracted by GP from different source domains and being used to a target domain. Specifically, we would like to investigate the following research objectives.

- Whether extracted features from source domains can be directly applied to target domains.
- Whether combining a set of features extracted by GP from source domains can compete with the features directly extracted by GP on a target domain.
- Whether increasing of the number of features extracted by GP from source domains and used for a target domain can improve test accuracy performance on the target domain.

1.2 Organisation

In the remainder of this paper, Sect. 2 briefly describes the background of document classification, transfer learning and GP for text classification. Section 3 introduces the proposed GP system to automatically extract features for document classification and directly apply extracted features to target domains. After Sect. 4 presents the design of the experiments, Sect. 5 provides the experimental results with discussions. Section 6 draws conclusions and addresses future work directions.

2 Background

2.1 Document Classification and Transfer Learning

The task of document classification is to classify a document to a predefined category based on the texts in the document. Statistical algorithms and artificial intelligence techniques have been used to automatically classify documents from provided datasets [1,14,19]. In document classification tasks, the frequency of words or the letter combination is considered as an important attribute to construct basic features. In general, the number of letter combinations are huge, and the frequency of each combination is not high. In the stage of pre-processing, some texts will be filtered, such as "and". In the process of feature extraction, it is popular to use n-gram techniques [1,3]. If we only consider the word specific combination, the bag-of-words model can be considered as a special case in the n-gram model. The number of occurrences of each word is a general and basic feature in a bag-of-words model. Considering the frequency of each word is not high and the set of words is large in a document classification task, we need effective feature selection techniques to select small sets of features from a high dimensional and sparse set of basic features. Traditional feature selection methods, such as information gain [21], are generally employed [1]. After a small set of features is selected, learning algorithms, such as Support Vector Machine (SVM) [13], are used to build classifiers.

In some document classification tasks, some topics (categories) are similar and usually include similar words. When the same input feature space is applied to these tasks and the distributions of the input features are different, the domain adaptation technique has been developed for similar topics [17]. When models (or features) are obtained from datasets which are considered as source domains, the models (or features) can be transferred to target domains including similar categories [18,20]. When training data is expensive or difficult to collect, we need an effective technique to obtain models (or features) with more easily obtained data from different domains. This technique is transfer learning.

In transfer learning, there are many types of methods for transferring knowledge from source domains to target domains [18,20]. Categorising transfer learning techniques mainly focuses on the difference between the source domain and the target domain. In a transfer learning task, when the input features in the source and target domains are different, the task is considered as heterogeneous transfer learning; otherwise it is homogeneous transfer learning. When a task considers the predefined labels in the source and target domains, it is considered as inductive transfer learning. Inductive transfer learning includes semi-supervised (only using a few labels) and supervised transfer learning. When the labels in the target domains are not provided, it is considered as transductive transfer learning. Unsupervised transfer learning is also used for transductive transfer learning. Since no labels are used in target domains, transductive transfer learning is very challenging [9,18,20].

In document classification tasks, there are n source domains SD, and each source domain SD_i $(i = 1, \ldots, n)$ has basic features X_{SD_i} and a marginal

probability distribution $P_{X_{SD_i}}$; and a target domain TD has a feature space X_{TD} and a marginal probability distribution $P_{X_{TD}}$. A subset feature space X_{sel,SD_i} from the source domain is selected by a feature selection algorithm; A model M_i is trained by using X_{sel,SD_i} to discriminate a document as category c. A category for a document is obtained by Eq. (1). Meanwhile, we can consider the model M_i as a high-level feature.

$$c = M_i(X_{sel,SD_i}) \tag{1}$$

Here, we only consider homogeneous transfer learning. In general, the source domain SD_i and the target domain TD have different marginal probability distributions. For the selected feature space X_{sel,SD_i}, SD_i has a marginal probability distribution $P_{X_{sel,SD_i}}$ and TD has a marginal probability distribution $P_{X_{sel,TD}}$. For a document from the source domain, its category c is obtained by Eq. (2). For a document from the target domain, its category c is obtained by Eq. (3) [17,20]. Here, TM (Transferred Model) is the model used in the target domain, which is generally driven from M. After $P_{X_{sel,TD_i}}$ is used to drive TM from M, Eq. (3) is adaptive to apply to the target domain.

$$c = M(X_{sel}, P_{X_{sel,SD_i}}) \tag{2}$$

$$c = TM(X_{sel}, P_{X_{sel,SD_i}}, P_{X_{sel,TD}}) \tag{3}$$

2.2 Related Work to GP for Text Classification

Based on the output and structure of GP programs, the categories of using GP for text classification are rule-based and value-based [6,10].

In a rule-based model (classifier), rules are easily understood by humans. So programs evolved by a rule-based GP system are normally readable. In [10,11], GP systems imitate the rule design from humans. In the GP systems, letters are used as terminals, and string operators and logical operators as functions. N-gram terms are generated by using an expand function and string connection function. Different from humans, the GP systems do not need an exhaustive search of all n-gram terms. GP will automatically find meaningful n-gram terms. However, since the evolved programs mainly focus on whether combinations of letters exist or not in a document, the frequencies of n-gram terms are not addressed, it is possible that complex and similar documents with different categories might not be handled well. Additionally, prior knowledge on rule-based text classification algorithms is required for setting up the GP system.

Similar to other value-based classification problem, a value-based GP system evolves programs to classify documents based on its output. In [6,22], predefined categories and existing features are provided to evolve binary GP programs. In the value-based GP systems, a small set of features are selected as terminals by feature selection algorithms. Common arithmetic operators, such as add, minus, times and division, are used to construct formulae. A threshold is used to discriminate whether a document belongs to a category or not based on the output of an evolved GP program. Different from a rule-based GP system, a

value-based GP system considers the frequencies of words and n-gram terms. Therefore, a value-based GP system can generally achieve better performance than a rule-based GP system [6,15,22].

3 The Method

This section introduces the proposed GP system for document classification used for transductive transfer learning. Here, we use a strongly typed and tree-based GP system. Firstly, we introduce the sets of terminals and functions, then describe the fitness function. Also, how to use composite features evolved by GP for another similar domain is introduced.

3.1 Functions and Terminals

Since GP has the ability to select features [7], traditional feature selection process is not used to select basic features in the proposed GP system. The numbers of the occurrences of words from a predefined vocabulary in a document are directly considered as terminals. If a word does not exist in a document, its occurrence will be 0. The proposed GP system randomly selects terminals to construct a GP tree. Therefore, the proposed GP system will automatically select features based on the predefined vocabulary. When the vocabulary are built from all training documents, the basic features from the vocabulary are the same to the features from the bag-of-words model. One advantage of using a vocabulary is that we can filter useless features, such as comma and full stop. Also, random constants are used in the GP system.

To construct a value-based GP tree, the function set contains four common arithmetic operators $\{+, -, \times, \div\}$. For example, we have two topics (categories) "food" and "motor". In order to classify a document from "food" or "motor", we have a GP program $\#[meal] - \#[wheel]$. Here, $\#[meal]$ and $\#[wheel]$ are the number of the occurrences of the word "meal" and the word "wheel"existing in the document. If the output of $\#[meal] - \#[wheel]$ is larger than threshold $th = 0$, the document is classified as label "food"; otherwise "motor".

In a binary document classification problem, there are positive label (class) "1" and negative label "0". A binary GP program from source domain SD_i in Eq. (4) classifies a document as "1" (the output of GP being larger than threshold th) or "0" (the output of GP not being larger than threshold th). Here c_i is the predicated label for the GP program evolved from source domain SD_i.

$$c_i = GP_{SD_i}(X_{SD_i}) > th?1 : 0 \tag{4}$$

Note that the training data in this paper is considered as balanced data, and only binary classification problems are addressed. The accuracy of correctly classified documents is used as the GP fitness function. Equation (5) defines the fitness function f, where N_T is the total number of correctly classified documents and N is the total number of documents.

$$f = \frac{N_T}{N} \tag{5}$$

3.2 Transductive Transfer Learning

GP evolves programs GP_{SD_i} from each source domain SD_i. However, the target domain TD is not totally the same to the source domain SD_i. The knowledge from a program evolved from SD_i might be partially helpful for predicating labels in the target domain. For the helpful part, it is considered as shared knowledge between source domain SD_i and the target domain TD. It is expected that GP evolved programs from source domain SD_i to transfer the shared knowledge to the target domain TD. To utilise the shared knowledge from GP evolved programs, we combine GP evolved programs from multiple source domains. Equation (6) describes a GP program $GP_{SD_i,j}$ to predicate a document in target domain TD as label $c_{i,j}$ ("0" or "1"). Here, j indicates one of GP programs evolved from source domain SD_i and $j = 1, \ldots, p$. Equation (7) describes that we randomly select p GP programs as extracted features from each source domain SD_i to combine as a final classifier for the target domain TD. The number of source domains is n.

$$c_{i,j} = GP_{SD_i,j}(X_{TD}) > th?1 : 0 \tag{6}$$

$$c = \sum\nolimits_{i=1,\ldots,n,j=1,\ldots,p} c_{i,j} > 0.5 * n * p?1 : 0 \tag{7}$$

To indicate the number of extracted features from each source domain, we use GP for the standard GP system (without transfer learning), GP_{vp} to indicate p GP programs as features from each source domain to vote for a document from the target domain as "1" or "0".

4 The Design of the Experiment

4.1 Dataset

The 20 newsgroup dataset [16] has been widely used by researchers [1,14,17]. The dataset includes 20 categories, and some are very closely related to each other, such as "rec.autos" and "rec.motorcycles". In "rec" group, there are four subcategories, namely "rec.autos", "rec.motorcycles", "rec.sport.baseball" and "rec.sport.hockey". Two groups "rec" and "talk" in the dataset are selected in this paper. Here, we consider one binary task of classifying "rec" and "talk" ("rec" vs talk"). "rec" is considered as label "1", and "talk"as label "0". Table 1 provides the details of the binary classification tasks. For instance, in "rec vs talk", when "rec.autos" and "talk.politics.guns" is a source domain, the others as the relevant target domains. $data_1$ is used to indicate the binary dataset which only includes "rec.autos" and "talk.politics.guns". The 20 newsgroup dataset has three different types of files. Here, "20news-bydate" is used. In "20news-bydate", its training dataset and test dataset are independent. Also, a vocabulary is provided in the 20 newsgroup dataset. The vocabulary includes 61188 words from the 20 newsgroup dataset.

Here, we take three datasets as source domains, and the other one as the target domain. Therefore, for each dataset, when it is the target domain, the other

Table 1. Four datasets used for source domain SD or target domain TD

	1	0
$data_1$	rec.autos	talk.politics.guns
$data_2$	rec.motorcycles	talk.politics.mideast
$data_3$	rec.sport.baseball	talk.politics.misc
$data_4$	rec.sport.hockey	talk.religion.misc

three are source domains. GP_{vp} automatically extracts features (programs) from the source domains and directly applies them to the test dataset on the target domain. It is assumed that some features are helpful to classify documents in the source domains and target domain. It is expected that GP will extract these helpful features from the source domains then these features can be effectively applied to the target domain.

4.2 GP Settings

We select the numbers of GP programs from each source as followings: 1, 3, 5. Namely, we have GP_{v1}, GP_{v3} and GP_{v5} to classify documents from the target domain.

Based on the common GP parameter settings in [8], the probability for mutation is 0.15, the probability for crossover is 0.80, the probability for elitism (reproduction) is 0.05, the population size is 200, and the maximum generations is 200. Since GP will automatically select features from a sparse and large set, the maximum depth (of a program) is 10, There are 30 independent runs for each experiment.

5 Results and Discussions

This section describes the test performance on each dataset with discussions. The results based on each dataset set as the target data and the others as the source datasets are provided. Further discussions are also presented.

5.1 Test Performance

Table 2 presents the test accuracies from GP, GP_{v1}, GP_{v3} and GP_{v5}. The fist row indicates the target domain. When one dataset is selected as the target domain, the other three datasets are the source domains. For instance, in the column "$data_1$", the source domains are $data_2$, $data_2$ and $data_3$, and the target domain is $data_1$. Here, we use two-pair sampled t-tests to compare the results from GP_{vp} to GP. The significance level 0.05 is used, and $\uparrow$ indicates that the test results from GP_{vp} are all significantly better than the results from GP. As we can see, only GP_{v1} on the target domain $data_2$ has non-significant results with GP; the others from GP_{vp} are significantly better than GP. From the four datasets, it

seems that the proposed GP system effectively extracts features from different source domains and effectively combines the extracted features as classifiers for similar target domains.

Table 2. Test accuracies on the target domain from GP, GP_{v1}, GP_{v3}, and GP_{v5}

	$data_1$	$data_2$	$data_3$	$data_4$
GP	0.7197 ± 0.0527	0.7516 ± 0.0447	0.7043 ± 0.0406	0.7389 ± 0.0444
GP_{v1}	0.7987 ± 0.0628 ↑	0.7339 ± 0.0954	0.7770 ± 0.0600 ↑	0.8014 ± 0.0806 ↑
GP_{v3}	0.8529 ± 0.0306 ↑	0.7812 ± 0.0503 ↑	0.8230 ± 0.0518 ↑	0.8552 ± 0.0478 ↑
GP_{v5}	0.8643 ± 0.0224 ↑	0.7880 ± 0.0337 ↑	0.8403 ± 0.0314 ↑	0.8675 ± 0.0315 ↑

There are also several interesting observations. Firstly, the test accuracies on $data_1$, $data_3$ and $data_4$ are improved more than 7%, comparing GP_{v1} with the standard GP method GP. Comparing the results from GP_{v3} and GP_{v5} with GP, the test accuracy improvement is more than 11% on datasets $data_1$, $data_3$ and $data_4$. Secondly, the test accuracy improvement on $data_2$ for GP_{vp} is not as large as the three datasets. It is possible that the three datasets $data_1$, $data_3$ and $data_4$ are close to each other. It is easier to find shared features for correctly classifying documents from the three datasets than $data_2$. Lastly, when the number of extracted programs is increasing, the test accuracy becomes higher; also, the standard deviations of the test accuracies become smaller. It seems that more features from the source domains can improve test performances on the accuracy and stability. More comparisons on the different numbers of features are given in the next subsection.

Figure 1 presents the details of the test accuracies of GP, GP_{v1}, GP_{v3} and GP_{v5} by box-plots.

First of all, when only one program evolved by GP from each source domain is combined together for a target domain, the test accuracy is not stable. Since these datasets do not have the same distribution on the input features, for a GP program evolved from a source domain, there might be limited shared knowledge between the source domain and a target domain.

Secondly, Fig. 1(2) shows one result with the test accuracy less than 0.5. It reveals that negative transfer occurs for the relevant combination. It is possible that the shared knowledge between a source domain and a target domain is not easily found. This paper only employs the voting strategy to classify documents. In order to avoid negative transfer when only single GP program is used for each source domain, the difference between the source domain and the target domain will be considered. Based on the difference, there are two potential ways to address the negative transfer issue. One way is to always to select evolved programs whose shared knowledge is easily found. The second way is to further handle with the evolved programs whose shared knowledge is not easily found so that the shared knowledge is easily used. Both ways are our future work directions.

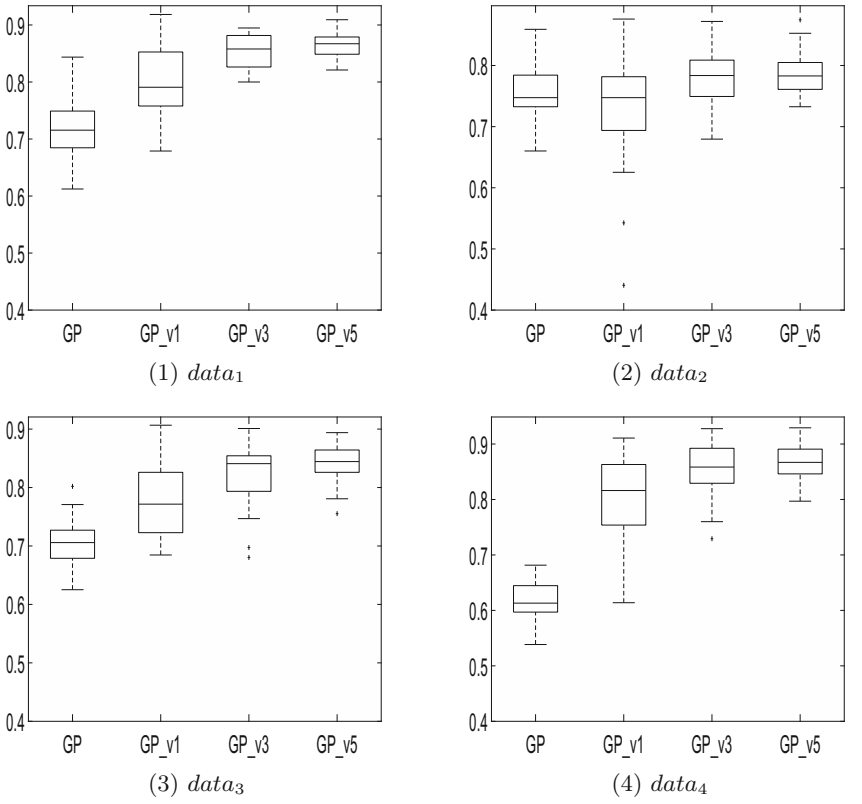

Fig. 1. Test Accuracies on the dataset from $data_1$ to $data_4$. In each figure, from the first to the fourth is the results from GP, GP_{v1}, GP_{v3}, and GP_{v5} respectively.

Thirdly, it also shows that the test accuracies are better when more the evolved GP programs are combined together for classifying documents from target domain. Since the evolved programs are different, there may also be difference of the shared knowledge from these programs. When more GP programs are used, the combination of the shared knowledge of these GP programs covers more for the dataset on the target domain. When the number of GP evolved programs reaches to a proper value, the coverage ratio of the knowledge on the target domain becomes stable. The experiments on the four datasets show that the test accuracy increment from GP_{v3} to GP_{v5} is obviously less than the increment from P_{v1} to GP_{v3}.

Fourthly, GP_{v3} and GP_{v5} do not have negative transfer based on the test accuracy on the four datasets. It seems that using multiple GP programs from each source domain can avoid negative transfer.

Finally, the best combinations can archive very high test performances. The results from the best combinations are higher than 0.9 on $data_1$, $data_3$ and $data_4$, and being close to 0.9 on $data_2$. It looks like that GP have potential to

automatically extract very good features. However, how to make GP to always generate very good features needs further investigation.

5.2 Comparisons Among GP_{v1}, GP_{v3} and GP_{v5}

Table 3 gives the multiple comparisons among GP_{v1}, GP_{v3} and GP_{v5} on the four test datasets. The multiple comparisons based on t-tests use Holm's method [12] for p-value adjustment, and the overall significance level is 0.05. "↑" means that the relevant row item is significantly better than the relevant column item. From the comparison results, GP_{v3} and GP_{v5} have significantly test accuracy results than GP_{v1}. However, there are no significant differences between GP_{v3} and GP_{v5}. From the multiple comparisons among GP_{v1}, GP_{v3} and GP_{v5} on the four test datasets, it also shows that multiple features (programs) evolved by GP from different source domains can improve test performance on the target domain than only using single GP features from source domains. When the number of GP features from each sourced domain used to a target domain reaches to a proper value, there is no significant improvement.

Table 3. Multiple Comparison Among GP_{v1}, GP_{v3} and GP_{v5} on Four Datasets $data_1$, $data_2$, $data_3$, and $data_4$

	GP_{v1}	GP_{v3}	GP_{v5}
$data_1$			
GP_{v3}	↑		
GP_{v5}	↑	–	
$data_2$			
GP_{v3}	↑		
GP_{v5}	↑	–	
$data_3$			
GP_{v3}	↑		
GP_{v5}	↑	–	
$data_4$			
GP_{v3}	↑		
GP_{v5}	↑	–	

5.3 Discussions

When a single feature from each source domain is applied to a target domain, negative transfer might occur. One evolved program which has negative transfer includes sub-tree $\#[car] - \#[gun]$. From its training data $data_1$, it is helpful to classify a document from "rec.auto" or "talk.politics.gun". However, "gun" may be not a general word for other data in group "talk". Since we do not consider

how to evaluate the difference between source domains and target domains, we do not know when negative transfer occurs. In [9], to obtain the same distribution for both domains, latent space construction aims to find the same distribution latent space between the feature spaces of the source and target domains. If the outputs of a GP feature is in a latent space, the distribution of the outputs of the GP feature may be helpful to check whether negative transfer occurs. In order to check whether a GP feature can be directly applied to a target domain, one potential way is to consider whether the outputs of the GP feature has a similar distribution to the raw source domain. This is our future work.

Also, the GP features are directly evolved from source domains without any information from a target domain. To make GP generate features which share knowledge between a source domain and a target domain, it is worth also considering the information from the target domain during an evolving stage. Evolved GP programs which are helpful for target domains include sub-trees, such as $\frac{\#[irregular]}{\#[technique]}$ and $\#[face] \times \#[improving]$. These words are general for groups "rec"and "talk". These general words are extracted by GP, and they are used as shared knowledge between source domains and target domains. Besides of the distributions of GP programs, it is worth to investigate how to integrate other information between the source and target domains into the GP fitness function.

6 Conclusions

This paper was to initially investigate transductive transfer learning in GP for automatically extract features for document classification based on four datasets. The GP evolved features from source domains were directly applied to a target domains. From the experimental results, when only a single GP evolved feature from each source domain is directly applied to the target domain, negative transfer might occur. However, from the overall results, the combinations of the GP evolved features from source domains have significantly better results on the target domain than the GP evolved single features directly from the target domain. Using multiple GP evolved features from each source domain can avoid negative transfer, also improves the test accuracy on the target domains, comparing the use of a single GP feature from the source domains to the target domain. More GP evolved features from each source domain to the target domain might not always significantly improve test accuracy.

In our future work, we will investigate how to avoid negative transfer. Also, the distribution difference between a source domain and a target domain will be considered into a GP fitness function so that evolved GP features can be adaptively applied to the target domain. Note that this paper only conducts investigation on the datasets being similar to each other. When source domains and a target domain are obviously different, how to effectively transfer the GP evolved features from the source domains to the target domain is future work.

References

1. Agarwal, B., Mittal, N.: Text classification using machine learning methods-a survey. In: Babu, B.V., Nagar, A., Deep, K., Pant, M., Bansal, J.C., Ray, K., Gupta, U. (eds.) Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28-30, 2012. AISC, vol. 236, pp. 701–709. Springer, New Delhi (2014). doi:10.1007/978-81-322-1602-5_75
2. Bhowan, U., McCloskey, D.J.: Genetic programming for feature selection and question-answer ranking in IBM watson. In: Machado, P., Heywood, M.I., McDermott, J., Castelli, M., García-Sánchez, P., Burelli, P., Risi, S., Sim, K. (eds.) EuroGP 2015. LNCS, vol. 9025, pp. 153–166. Springer, Cham (2015). doi:10.1007/978-3-319-16501-1_13
3. Cavnar, W.B., Trenkle, J.M.: N-gram-based text categorization. In: Proceedings of 3rd Annual Symposium on Document Analysis and Information Retrieval, SDAIR-1994, pp. 161–175 (1994)
4. Chen, Q., Xue, B., Niu, B., Zhang, M.: Improving generalisation of genetic programming for high-dimensional symbolic regression with feature selection. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp. 3793–3800 (2016)
5. Chen, Q., Zhang, M., Xue, B.: Feature selection to improve generalisation of genetic programming for high-dimensional symbolic regression. IEEE Trans. Evol. Comput. **PP**(99), 1 (2017)
6. Escalante, H.J., García-Limón, M.A., Morales-Reyes, A., Graff, M., Montes-y-Gómez, M., Morales, E.F., Martínez-Carranza, J.: Term-weighting learning via genetic programming for text classification. Knowl.-Based Syst. **83**, 176–189 (2015)
7. Fu, W., Johnston, M., Zhang, M.: Low-level feature extraction for edge detection using genetic programming. IEEE Trans. Cybern. **44**(8), 1459–1472 (2014)
8. Fu, W., Johnston, M., Zhang, M.: Distribution-based invariant feature construction using genetic programming for edge detection. Soft Comput. **19**(8), 2371–2389 (2015)
9. Gong, B., Grauman, K., Sha, F.: Learning kernels for unsupervised domain adaptation with applications to visual object recognition. Int. J. Comput. Vis. **109**(1), 3–27 (2014)
10. Hirsch, L., Saeedi, M., Hirsch, R.: Evolving rules for document classification. In: Keijzer, M., Tettamanzi, A., Collet, P., van Hemert, J., Tomassini, M. (eds.) EuroGP 2005. LNCS, vol. 3447, pp. 85–95. Springer, Heidelberg (2005). doi:10.1007/978-3-540-31989-4_8
11. Hirsch, L., Saeedi, M., Hirsch, R.: Evolving text classification rules with genetic programming. Appl. Artif. Intell. **19**(7), 659–676 (2005)
12. Holm, S.: A simple sequentially rejective multiple test procedure. Scand. J. Stat. **6**(2), 65–70 (1979)
13. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998). doi:10.1007/BFb0026683
14. Khan, A., Baharudin, B., Lee, L.H., Khan, K., Tronoh, U.T.P.: A review of machine learning algorithms for text-documents classification. J. Adv. Inf. Technol. (2010)
15. Khodadi, I., Abadeh, M.S.: Genetic programming-based feature learning for question answering. Inf. Process. Manage. **52**(2), 340–357 (2016)
16. Lang, K.: Newsweeder: learning to filter netnews. In: Proceedings of the 12th International Machine Learning Conference (ML95) (1995)

17. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE Trans. Neural Netw. **22**(2), 199–210 (2011)
18. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 1345–1359 (2010)
19. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. **34**(1), 1–47 (2002)
20. Weiss, K., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. J. Big Data **3**(1), 9 (2016)
21. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proceedings of the Fourteenth International Conference on Machine Learning, pp. 412–420 (1997)
22. Zhang, B., Fan, W., Chen, Y., Fox, E.A., Gonçalves, M.A., Cristo, M., Calado, P.: A genetic programming approach for combining structural and citation-based evidence for text classification in web digital libraries. In: Herrera-Viedma, E., Pasi, G., Crestani, F. (eds.) Soft Computing in Web Information Retrieval. Studies in Fuzziness and Soft Computing, vol. 197, pp. 65–83. Springer, Heidelberg (2006). doi:10.1007/3-540-31590-X_4