ISBN: 978-981-18-1791-5
2021 the 11th International Workshop on Computer Science and Engineering (WCSE 2021)
doi: 10.18178/wcse.2021.06.011

GraftNet: An Efficient and Flexible Multi-label Image Classification and Its Application

Chunhua Jia[1] , Shuai Zhu [1+], Wenhai Yi[1], Yu Wu[1], Leilei Wu[1], WeiweiCai[1]

[1] Shanghai Elevator Media Information Co., Ltd., China

Abstract. A multi-label network with an adaptive hierarchical structure is presented in this paper. The proposed network is efficient and flexible; we call it GraftNet in abbreviation. The network is in a tree-like fashion which has a trunk and several branches. The trunk is pre-trained with a dynamic graph for generic feature extraction and branches separately trained on sub-datasets with single label to improve accuracy and efficiency. We employ efficient neural architecture search (ENAS) for the branches part which outperforms manually designed architectures on image classification. The proposed network could avoid the inherent problem of "catastrophic forgetting" problem in deep neural networks and the addition of new image classes to a network could be easier than retraining the whole network again with all the classes. The employment of ENAS helps our network to achieve high performance. The proposed hierarchical network is also suitable for imbalanced dataset. Compared against fine-tuning a deep network, the proposed network achieves significant reduction of training and testing effort. Experimental results show that it has good performance on our human attributes recognition task.

Keywords: Convolutional Neural Networks· Deep Learning· Incremental Learning· Transfer Learning · NAS.

1. Introduction

Deep Convolutional Neural Networks (DCNNs) have emerged as the leading architecture for large scale image classification and have demonstrated promising performance in single-label image classification tasks [1]. Deep convolutional neural network (AlexNet) was proposed to classify the 1.2 million high-resolution images into the 1000 different classes [2]. An improved Convolutional Neural Network structure was proposed whose first layer of the network structure uses convolutional layer inversion to increase the transmission of effective feature information. A new sum-product network structure combining with the novel image feature extraction method was proposed for flower image classification [3]. A fruit image classification algorithm based on deep learning and migration learning under TensorFlow framework was proposed. The partial model structure of Inception-V3 was used to extract the feature of fruit image and softmax classifier was used to classify the image features [4].

One intuitive method is to transfer multi-label task into a set of binary classifications for each single label [5]. It is redundant to repeatedly extract some general features, which can probably be extracted once with shared model. Although this solution is optimized by techniques such as regularization and label encoding [6], it is still not straightforward. One other solution uses a transfer-learned network to extract features, then, independent classifiers such as SVM are applied to predict for each label. It is often explored and utilized in multi-label classification in past years [7]. CNN is a much more powerful feature extractor, by combining off-the-shelf CNN with SVM the computation cost can be reduced, but problems still exist, as CNN-SVM is a two-step process and is less accurate compared to full CNN models. In addition, there is a more direct network that is end-to-end and treats multi-label prediction as one complete task [8]. In this solution, tree-like network is a typical architecture that uses shallow layers to extract low-level common features and then uses separate branches to extract high-level features for each label [9]. Because of its good classification performance and high computational efficiency, tree-like networks are much used in recent days.

+ Corresponding author.
E-mail address: zhushuaisjtu@qq.com

However, tree-like networks are designed for a static dataset with fixed number of labels and can not effectively deal with the situation where the labels increase dynamically. It is difficult to response for new labels. It consumes too much resource in annotation and training for the additional labels. When adding new labels. We need to annotate not only the new images collected for this new label but also all the existing images to identify whether they belong to this new label or not. As we cannot know the labels to be predicted in advance, the whole dataset must be re-annotated again and again for additional labels. When the number of labels keeps increasing and the dataset increases to hundreds thousands or millions level, which costs much labour and time. Besides, training on the whole dataset all over again is also costly. Class imbalance is a crucial problem in machine learning, the classification model tends to favour the majority classes and incorrectly classify instances from the minority classes as belonging to the majority classes, leading to poor predictive accuracies. When trained on one task, then trained on a second task, many machine learning models "forget" how to perform the first task. "Catastrophic forgetting" is widely believed to be a serious problem for neural networks [10]. Meanwhile, previous multi-label image classification is largely limited by the representation power of the hand-crafted features, which is a time-consuming and error-prone process.

2. Related Works

Target of multi-label classification is to predict multiple labels for each input image, which is usually achieved by transforming multi-label problem into a set of single label problems. The basic technique is binary relevance, which trains one binary classification model for each label [1]. The parameter sharing method [11] is similar to tree model. More concretely, AlexNet which is pretrained on the ImageNet was used to extract features and SVM was used to train for each label, the network was fine-tuned by target multi-label dataset to learn more domain-specific features.

While deep convolutional neural networks (CNNs) have shown a great success in single-label image classification, it is important to note that real world images generally contain multiple labels. Multi-label network classification is a well-known task that is being used in a wide variety of web-based and non-webbased domains [12] which has achieved great progress due to the development of deep convolutional networks [13]. Although how DCNN best copes with multi-label images still remains an open problem, it has been successfully applied into many real-world applications [14]. CNN-RNN framework was proposed which combined recurrent neural networks (RNNs) and CNNs to learn a joint image-label embedding to characterize the semantic label dependency as well as the image-label relevance. And it can be trained end-to-end from scratch to integrate both information in an unified framework [15]. A flexible deep CNN infrastructure [16] was proposed where an arbitrary number of object segment hypotheses were taken as the inputs, then a shared CNN was connected with each hypothesis and finally the CNN output results from different hypotheses were aggregated with max pooling to produce the ultimate multi-label predictions. A global/local attention method was presented that can recognize a multi-label image from coarse to fine by mimicking how human-beings observe images. A joint max-margin objective function was introduced, which enforces that the minimum score of positive labels should be larger than the maximum score of negative labels horizontally and vertically [17].

A DCNN embeds feature extraction and classification in one coherent architecture within the same model [18]. As a building block, it is now well positioned to be part of a larger system and modifying one part of the parameter space immediately affects the model globally. An unresolved issue is that such model is trained on a static snapshot of data. Hybrid sampling method [19] was used to balance the number of training examples in such a multi-class setting. For neural networks, "catastrophic forgetting" is a problem faced by many machine learning models and algorithms. When trained on one task, then trained on a second task, many machine learning models "forget" how to perform the first task [10]. Due to "catastrophic forgetting", adding new labels for multi-label classification has to relabel the train data and retrain the whole model, which is time-consuming and inefficient. Most algorithms on multi-label classification are designed for balanced data. SMOTE is a technique that has been applied for balancing single-labeled as well as multi-label frameworks [20].

In order to effectively train multi-label model, end-to-end network [21] emerged that learned features from shared layers and several task-specific branches. On the other hand, for the purpose of refining

predicted results, many efforts in multi-label classifications area were put into capturing label relations. Ranking-based learning strategy that transforms the multi-label problem into label ranking problem and label space encoding. Then, regarding the predicted result of last label as the input embedding of next model, were designed to consider correlations among labels so that performance of multi-label classification can be improved when intra-class information exists [22].

Multi-label learning deals with problems where a single instance while being associated with multiple class labels simultaneously [23]. It is redundant to repeatedly extract some general features which can probably be extracted many times. Motivated by the advances of compressive sensing, a sparse representation that selects a compact subset to describe the input data can be more discriminative. A sparse multi-label learning method to circumvent the visually polysemous barrier of multiple tags [7]. The hierarchy relationship between coarse classes and fine classes was discussed to achieve a better performance on fine-grained classification and meanwhile reduce the heavy dependence on fine labels. A global/local attention method was introduced to recognize an image from coarse to fine. Attention models designed for sequence prediction such as recurrent neural networks (RNNs), long short-term memory (LSTM) and so on. The model were used to encode labels into embedding features and thus making the relationship among labels better exploitable [15]. Hierarchy and Exclusion (HEX) graph was involved in exploring correlations of labels, which captures semantic relations by using mutual exclusion, overlap and subsumption [24]. Nevertheless, multi-label classification is not only used to describe whole image attribute, but also used to describe local attribute. In order to extract local features, objects proposals [25] that includes possible foregrounds were applied.

Previous Multi-label image classification is largely limited by the representation power of the hand-crafted features, which is a time-consuming and error-prone process. Neural architecture search (NAS) aims to discover network architectures with desired properties such as high accuracy or low latency. Because of this, there is growing interest in automated neural architecture search methods [26]. A recurrent network to generate the model descriptions of neural networks and train this RNN with reinforcement learning to maximize the expected accuracy of the generated architectures on a validation set. A fast and inexpensive approach for automatic model were designed [27]. An efficient neural architecture search method [28] shows that a simple yet powerful evolutionary algorithm can discover new architectures with excellent performance. The approach combines a novel hierarchical genetic representation scheme that imitates the modularized design pattern commonly adopted by human experts, and an expressive search space that supports complex topologies. An unified framework for NAS was presented, which encapsulates recent DNAS and RL-based approaches under one framework [29].

It still takes long time and lots of resources to train a complete deep neural network for multi-label image classification. In this paper, an efficient and flexible multi-label image classification we call it GraftNet is proposed, which has a trunk and several branches. GraftNet is resource-saving and flexible, especially for incremental extension on new labels. Specifically we use a dynamic data graph to train trunk of our model, which solves problems existing in annotating and training. GraftNet is also motivated by UberNet [30] that jointly handles low-, mid-, and high-level vision tasks in a unified architecture.

The rest of the paper is organized as follows. We discuss the model structure, training details and its application in Section 3. In this section, we introduce the training dataset firstly, then, we describe how to train the trunk and how to build an efficient branches. Some training tricks and the application are also discussed in this section. Experiments are shown in Section 4.

3. GraftNet

GraftNet consists of a trunk and several branches, but they are trained separately and both of them are trained on a set of sub-datasets. These sub-datasets are collected for each attribute and each of the sub-datasets only contains one label. The purpose of training the trunk is to make the trunk more adaptive and generic to all attributes. We propose a dynamic graph to train the trunk on the sub-datasets jointly. The branches are simply trained on its corresponding sub-dataset separately, which based on the pre-trained trunk. After training, the branches are grafted to the trunk. The trunk is trained once time and a new branch is

trained and grafted immediately whenever a new attribute is added. As for the branches, Efficient Neural Architecture Search (ENAS) is used for an optimal subgraph.

3.1. Sub-datasets

As mentioned above, when we add a new label to a dataset, we not only need to annotate the new images but also have to re-annotate the whole dataset. When the dataset is increased to a large size and new labels are added constantly, this process will waste much time. In our method, GraftNet is trained on some small sub-datasets. Instead of maintaining a whole dataset, we only collect some small independent sub-datasets for each attribute. When a new label is added, we just need to collect and annotate images for this new label, this will not influence the previous sub-datasets. For N labels there will be N sub-datasets, which saves much cost on annotating the previous images. GraftNet is mainly used for human multiple attributes classification; we collect the images from real elevator scenarios and use instance segmentation to segment the people masks. Compared with object detection, instance segmentation could remove the background as much as possible. Considering the number of images needed to be processed is very large, one-stage instance segmentation method YOLACT is chosen to segment the images. For each attribute, the segmented people masks are collected to build the sub-dataset. We use all the sub-datasets to train one trunk and each of the sub-datasets to train a independent branch. As traditional training method does not work when training on all the sub-datasets, a dynamic graph is proposed to train the trunk on all the sub-datasets jointly.

3.2. The Trunk: Pre-trained Weights by Using Dynamic Graph

Transfer learning is a common way to train model for a new task, which helps to accelerate training convergence and achieves better accuracy, especially when the target dataset is relatively small. Usually the pretrained weights for transfer-learning are gathered from the training on some public datasets, e.g. ImageNet, COCO, etc., which is tremendously large and thus the weights are generic for many tasks. Recently, research [31] pointed out that transfer-learning benefits more from pretraining on a source domain that is more similar to target domain. In our case, target datasets are bunches of images of one person with one attribute labeled. Clearly there is low domain similarity between target datasets and public datasets such as ImageNet, if we finetune the branches based on the public pretrained weights directly it will performs worse because of low domain similarity. In our method we select a part of sub-datasets and combine them to act as the source domain, and train one trunk based on this dataset to get a set of pretrained weights. Then each branch is to finetune based on this trunk on their own sub-dataset. This could make the domain similarity high enough because the selected sub-datasets are actually the same type with the target sub-dataset.

As shown in Fig. 1, to increase the domain similarity between the trunk and the targets, we design a graph with dynamic data flow in pre-training stage based on the selected sub-datasets directly. Besides, as we use the trunk to extract common features among all the attributes and then use the branches to predict each attribute accurately. We want the trunk to be much generic to all the attributes. Specifically, the dynamic data flow graph consists of the Conv layers, the Batch Normalization (BN) layers, a Global Average Pooling (GAP) layer and a customized fully connected layer. The customized FC layer contains several sets of weights and biases. For N attributes it is initialized with N sets of weights and biases. During a training step, an attribute index is randomly selected, and a batch of images and labels are fetched from the sub-datasets with the same index. This batch of images is sent to the Conv layers, BN layers and GAP layer to extract features. Then one set of weights and biases is selected and activated according to the attribute index to form a standard FC layer. This selected FC layer continues to process the feature maps and we use a softmax activation to obtain the predicted vector. Finally the loss is computed based on the selected labels and the model is updated. In next training step, a new attribute index is randomly selected and the whole process will repeat again. For each training step, the set of weights and biases of the FC layer is dynamically changing according to the attribute index. By using this kind of dynamic graph, the Conv layers are updated for all the attributes and the FC layer is only undated for a specific attribute. This makes the Conv layers as much adaptive and generic to all attributes as possible. After training only the weights of Conv layers and Batch Normalization (BN) layers will be saved as trunk for the fine-tuning of each branch and the FC layers will be excluded.

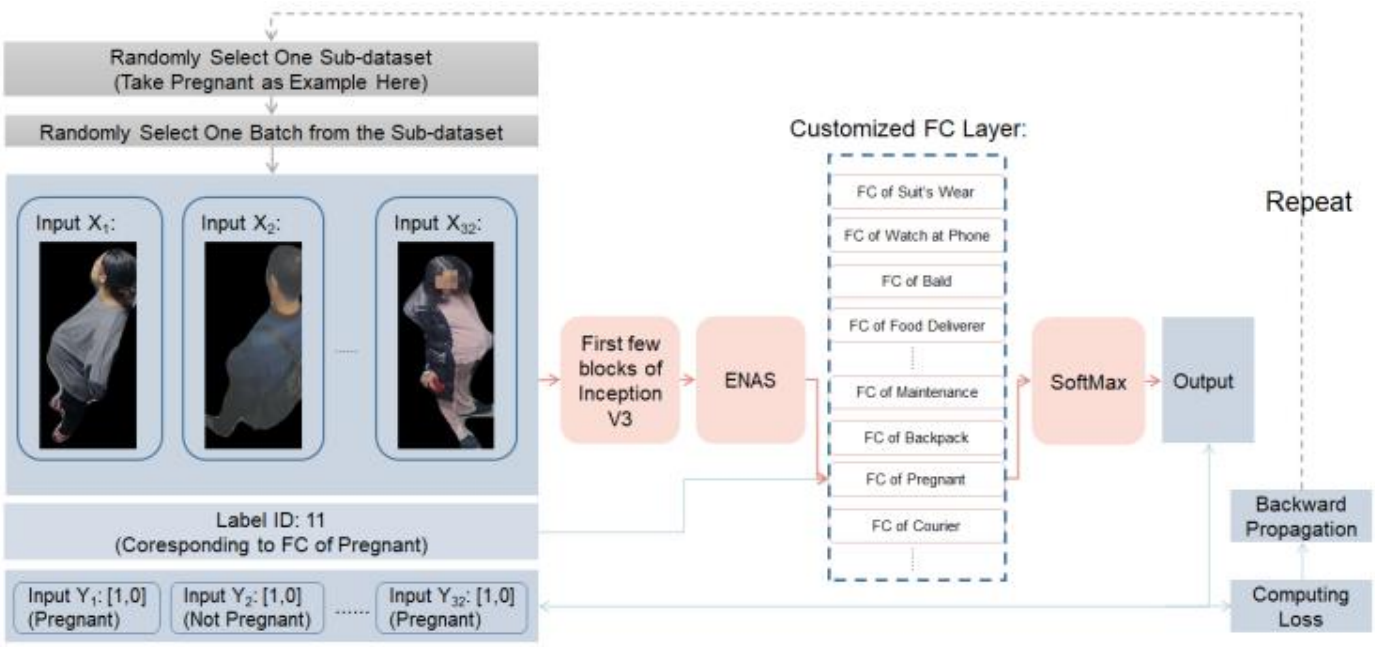

Fig.1: Process of training with dynamic data flow graph.

The loss function of this dynamic graph network is written as below:

$$Loss = \sum_{t=0}^{T} \varepsilon_t L_t(G_{w_0,w_t}, y_t) \tag{1}$$

where $\boldsymbol{t}$ represents the index of attributes as shown in Fig. 1, $\boldsymbol{w_0, w_t}$ represent the weights of InceptionV3 trunk and customized FC layer, $\boldsymbol{G_{w_0,w_t}, y_t}$ represent the attribute-specific network prediction and the corresponding ground truth, and $\boldsymbol{\varepsilon_t} \in \{\boldsymbol{0}, \boldsymbol{1}\}$ represents whether the input samples belong to the sub-dataset of one attribute. To simplify the training procedure, one batch of samples is randomly picked from one of sub-datasets.

3.3. The Optimal Structure for Each Branch

The benefit of ENAS is that we can use in this subsection to balance the needs of reducing model complexity and ensuring actual effect. According to the heavy burden of training large numbers of parameters of inception V3 model, we would like to optimize the whole training progress. Borrowing the idea of Fine-Tuning, we use the output feature map of first seven blocks as the model input and try to replace the last three blocks of inception V3 model with our architecture in an effective and light way.

At first, we define network search space which includes the choose of activation layer of each node and existence of the edge between each pair of nodes. We use LSTM as our controller network to build and train our network sampler and trainer to find the best architecture. The learnable parameters include two parts: the parameters of the controller LSTM, the shared parameters of the child models. We use interleaving to train these two sets of parameters and finally choose a well-trained network sampler according to test accuracy of each epoch. For all sampled network architectures of the well-trained sampler, we choose the best network architecture which achieves best performance in test dataset. This architecture is the final output of whole search process. Then, according to the output of above step, we construct and train our own network. The quantifiable outcomes of different tasks show that the architecture searched by ENAS achieves a slightly better result as raw inception v3 architecture with much fewer parameters. For each branch, ENAS is employed to achieve the best architecture, as show in Fig.2.

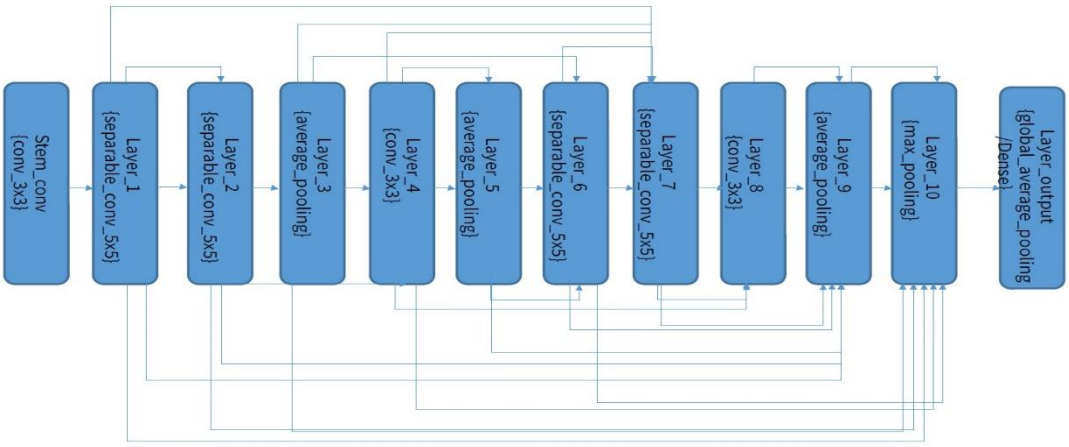

Fig.2: The best architecture result of macro search progress.

3.4. Architecture Details

To recognize human attributes, low-level features such as shapes and textures could be commonly represented with generic features and high-level features should be customized for different attributes, therefore GraftNet is designed as shown in Fig. 3. We use InceptionV3 to train and construct GraftNet on the sub-datasets by using a two-step training procedure. In first step, the InceptionV3 is trained jointly on the collection of the sub-datasets by the proposed dynamic data flow graph (InceptionV3 consists of 11 blocks, in our paper all our grafting operations are block basis). After training the Conv blocks with pretrained weights could be considered as the trunk of GraftNet. The second step is to fine-tune and graft the branches to the trunk for each attribute. As shown in Fig. 3, the trunk is for low-level generic feature extraction and is shared with all branches during inference. The branches extract high-level features specifically for each attribute. To simplify dataset management and reduce cost, we collect and annotate sub-datasets for single attribute only, branches are trained on sub-datasets correspondingly and then grafted onto the trunk. Regarding the structure of branches, for most of them we simply follow the original structure of InceptionV3, meaning the fine-tuning is based on the InceptionV3 with the pre-trained weights of the trunk, we only have to decide how many consecutive blocks to be set as trainable according to the actual result. More blocks can be opened for fine-tuning if the attribute is difficult to recognize. ENAS is utilized for each branch, which is discussed in subsection 3.3. GraftNet is deployed on cloud and it is flexible to extend some new labels, and it is efficient once the best neural architecture is employed.

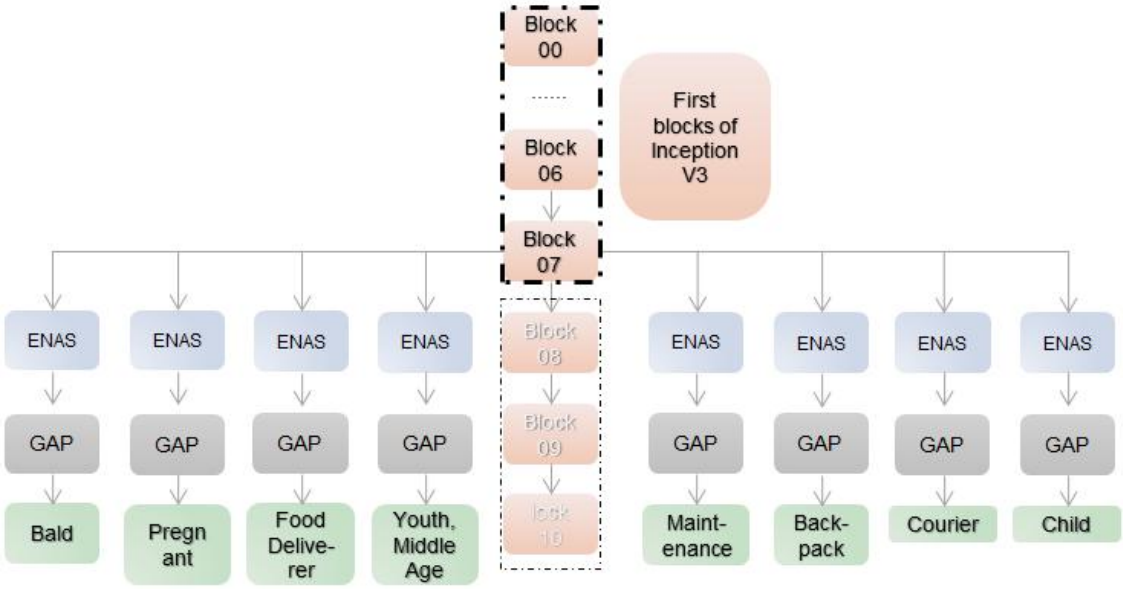

Fig.3: Architecture of GraftNet.

3.5. Training Frequency of Trunk and Branches

As the trunk is trained jointly on the collection of the sub-datasets by dynamic data flow graph, it usually costs too much time for every training. To reduce cost, the trunk is trained once after 5−7 new labels are accumulated. At the beginning we only had 5 labels, when the number of labels increased to 10, we re-train the trunk on the 10 sub-datasets and update the branches. As for branches, a new label means a new branch, which is trained on the corresponding sub-dataset.

3.6. Application on Advertising

A prototype of real-time advertising system is built which is partly based on our implementation of elevator passenger attributes recognition. LED screens and cameras are both mounted in each elevator with internet access. YOLACT and GraftNet are deployed on cloud to capture attributes of elevator passengers from uploaded camera snapshot and then advertisements can be pushed onto LED screen accordingly.

4. Experiments

4.1. Dataset and Experiment Setup

We collect independent sub-datasets for each attribute from real scenarios, including positive samples and negative samples. All the images are collected via the cameras installed in 120 thousand elevators. We use these cameras to get the video screenshots of the elevators and upload the screenshots to a web database. The resolution of the screenshots is 1920×1080 or 1280×720, and the downloaded screenshots are then processed by YOLACT to segment the samples for training and testing. These segmented samples are manually selected and are used to make up each independent dataset. The numbers of images of each attribute we collected is shown in Table. 1. As shown in the table, different categories have different numbers of samples which are imbalance datasets. As show in Table. 1, millions of images are collected. Then the trunk is trained on these sub-datasets jointly by the proposed dynamic graph, and then find the best architecture for each sub-dataset, separately. As we only need to train on this independent sub-dataset, each of them only has tens of thousands of images and it is much easier to collect and annotate.

Table 1. Number of Positive and Negative Images For Each Attribute.

Attributes	Number of train images: positive/negative	Number of test images: positive/negative
Suit's Wear	20876/96092	2486/9998
Watching Phone	116531/199291	1226/19582
Bald	57511/116619	1105/5957
Food Deliverer	56647/143042	5382/10658
With Fashion Bag	14713/80263	672/19328
Youth	84330/50833	10523/16774
Middle Age	85815/49348	6552/20745
Holding Baby	27907/187138	2287/79205
Sporty	12983/91529	5433/57303
Maintenance	26181/86701	10323/22173
Backpack	28697/81504	7400/40000
Pregnant	8131/106533	903/11837
Courier	26181/86701	4768/18173
Old Age	99347/525590	23889/127822
Baby	46348/166021	2707/79205
Male	180275/177706	47636/48344
Female	192147/165834	37909/58071
Child	110147/510938	31770/119927

4.2. Training Details

The training process includes training the trunk and fine-tuning the branches. We first use dynamic graph to train an InceptionV3 to get a base InceptionV3 network. The base InceptionV3 will act as trunk for all the branches. Then we start to train the branches on the small sub-datasets. For this step of training process, we

first load the pre-trained weights of the previous trained baseInceptionV3, freeze specific number of blocks and get the features by inference of the frozen inception V3 model. Then, ENAS is implemented for the best architecture based on the features which is described in subsection 3.3. Adding new categories means only the branches need to be trained. The trunk will be trained once or twice a month and the training process usually takes 3 4 days for dynamical engineering situation. The new InceptionV3 trained on small sub-dataset will be trained when new demand comes and it usually takes 1 day. This training manner helps us in quickly responding to new demands.

4.3. Results Analysis

Experimental results of the collected datasets are shown in the Table. 2, the accuracy of test stage for each attribute is computed. We first analyze the overall performance of GraftNet trained with graph network pre-trained weights, and then compare it with the version trained with ImageNet pre-trained weights to verify the proposed graph network procedure. As shown in Table. 2, we take 6 attributes in our experiment as example, they are Child, Pregnant, Bald, Maintenance, Courier and Food Deliverer. Comparing to Table. 1, the positive sample number for training of Child is 110147 while Pregnant only has 8131 samples. As shown in the table, the first row the accuracy GraftNet and the second row is weight parameter reduction rate, compared to Inception V3 model. GraftNet could dramatically reduce the number of parameter without influence the model accuracy when comparing with Inception V3.

Table 2. Comparison of accuracy and weight number.

Attributes	Child	Pregnant	Bald	Maintenance	Courier	Food Deliverer
Accuracy	97.1%	94.4%	95.0%	89.4%	91.8%	92.7%
Reduction Rate	-65.6%	-76.5%	-88.5%	-81.3%	-88.7%	-85.8%

5. Conclusion

In this paper, we have proposed GraftNet whose trunk extracts low-level features and branches extract high-level features. GraftNet is suitable for fine-grained multi-label classification task in industry implementation. The proposed network is adaptive hierarchical which is flexible and extendable for adding new labels without worrying about the “catastrophic forgetting” problem. With the help of pre-trained trunk, there is no need large number of annotations. Meanwhile, it is suitable for imbalanced dataset with different categories. Especially, ENAS is utilized for the best neural architecture of each branch to extract customized high-level features to corresponding attributes, which is efficient for inference. GraftNet is able to reduce time spent on annotation and training, simplify multi-label classification. GraftNet enhances speed of reaction to predicting new labels and maintain relatively high accuracy of fine-grained tasks. Detailed experimental results prove the ascendancy of our framework.

6. References

[1] Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: A comprehensive review. Neural Computation p. 1 (2017)

[2] Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems 25(2) (2012)

[3] Xiaojie, S., Youlong, Y.: Flower image classification research based on sum-product networks. Computer Engineering and Applications (2019)

[4] Xiao-Qin, L., Kai-Yuan, C., Sa, A.N., Ye-Lan, W.U., Wen-Yang, G.: Fruit image classification based on deep learning and transfer learning. Measurement & Control Technology (2019)

[5] Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. International Journal of Data Warehousing and Mining 3(3), 1–13 (2007)

[6] Duong, L., Cohn, T., Bird, S., Cook, P.: Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In: Meeting of the Association for Computational Linguistics & International Joint Conference on Natural Language Processing (2015)

[7] Han, Y., Wu, F., Zhuang, Y., He, X.: Multi-label transfer learning with sparse representation. IEEE Transactions

on Circuits and Systems for Video Technology 20(8), 1110–1121 (2010)

[8] Aljundi, R., Chakravarty, P., Tuytelaars, T.: Expert gate: Lifelong learning with a network of experts (2016)

[9] Kontschieder, P., Fiterau, M., Criminisi, A., Bulo, S.R.: Deep neural decision forests. In: IEEE International Conference on Computer Vision (2016)

[10] Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks. Computer ence 84(12), 1387–91 (2013)

[11] Seltzer, M.L., Droppo, J.: Multi-task learning in deep neural networks for improved phoneme recognition (2013)

[12] Rashed, A., Grabocka, J., Schmidt-Thieme, L.: Multi-label network classification via weighted personalized factorizations (2019)

[13] BOUTELL, M: Learning multi-label scene classification*1. Pattern Recognition (2004)

[14] Devkar, R., Shiravale, S.: A survey on multi-label classification for images. International Journal of Computer Applications 162(8), 39–42 (2017)

[15] Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., Xu, W.: Cnn-rnn: A unified framework for multi-label image classification. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2285–2294 (2016)

[16] Wei, Y., Xia, W., Lin, M., Huang, J., Ni, B., Dong, J., Zhao, Y., Yan, S.: Hcp: A flexible cnn framework for multi-label image classification. IEEE Transactions on Software Engineering 38(9), 1901–1907 (2016)

[17] Fan, L., Li, L., Victor, S.S., Fu, Q., Hu, F.: Multi-label image classification via coarse-to-fine attention*. Chinese Journal of Electronics 28(6), 1118–1126 (2019)

[18] Roy, D., Panda, P., Roy, K.: Tree-cnn: A hierarchical deep convolutional neural network for incremental learning. Neural Networks (2018)

[19] Agrawal, A., Viktor, H.L., Paquet, E.: Scut: Multi-class imbalanced data classifi-cation using smote and cluster-based undersampling. In: International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management

[20] Giraldo-Forero, A.F., Jaramillo-Garzn, J.A., Ruiz-Muoz, J.F.: Managing imbalanced data sets in multi-label problems: A case study with the smote algorithm. (2013)

[21] Razavian, A.S., Azizpour, H., Sullivan, J., Carlsson, S.: Cnn features off-the-shelf: an astounding baseline for recognition (2014)

[22] Yeh, C.K., Wu, W.C., Ko, W.J., Wang, Y.C.F.: Learning deep latent spaces for multi-label classification (2017)

[23] Zhang, M.L., Yukun, L.I., Liu, X.Y., Geng, X.: Binary relevance for multi-label learning: an overview. Frontiers of Computer ence 12(2) (2018)

[24] Jia, D., Nan, D., Jia, Y., Frome, A., Adam, H.: Large-scale object classification using label relation graphs. In: European Conference on Computer Vision (2014)

[25] Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis & Machine Intelligence 39(6), 1137–1149 (2017)

[26] Elsken, T., Hendrik Metzen, J., Hutter, F.: Neural architecture search: A survey. arXiv (2018)

[27] Dong, X., Yang, Y.: Searching for a robust neural architecture in four gpu hours (2019)

[28] Liu, H., Simonyan, K., Vinyals, O., Fernando, C., Kavukcuoglu, K.: Hierarchical representations for efficient architecture search (2017)

[29] Vahdat, A., Mallya, A., Liu, M.Y., Kautz, J.: Unas: Differentiable architecture search meets reinforcement learning. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)

[30] Kokkinos, I.: Ubernet: Training a 'universal' convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In: 30th IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[31] Cui, Y., Song, Y., Sun, C., Howard, A., Belongie, S.: Large scale fine-grained categorization and domain-specific transfer learning. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)