Interspeech 2018
2-6 September 2018, Hyderabad

Adding New Classes Without Access to the Original Training Data with Applications to Language Identification

Hagai Taitelbaum, Ehud Ben-Reuven, Jacob Goldberger

Engineering Faculty, Bar-Ilan University, Israel

hagai.taitelban@live.biu.ac.il, udi.benreuven@gmail.com, jacob.goldberger@biu.ac.il

Abstract

In this study we address the problem of adding new classes to an existing neural network classifier. We assume that new training data with the new classes is available. In many applications, dataset used to train machine learning algorithms contain confidential information that cannot be accessed during the process of extending the class set. We propose a method for training an extended class-set classifier using only examples with labels from the new classes while avoiding the problem of forgetting the original classes. This incremental training method is applied to the problem of language identification. We report results on the 50 languages NIST 2015 dataset where we were able to classify all the languages even though only part of the classes was available during the first training phase and the other languages were only available during the second phase.

Index Terms: language identification, adding new classes, catastrophic forgetting, learning privacy.

1. Introduction

Assume we are given a deep neural network that was trained to classify an object into one of the classes in a pre-defined set. Often, our specific task requires adding new classes and given an instance we need to select a class from the extended class-set. We are given training data points that are labeled by categories from the new set and our goal is to extend the given network classifier to handle both the original and the new classes. The obvious solution is to start training a classification network from scratch that can classify an instance to one of the classes in the extended set.

In this study we address a problematic but a common situation where we cannot access the training data points that were used to train the classification system (but we can inspect the parameters of the trained model). We can only use a new dataset that solely contains training examples with labels from the new category set. This situation can occur when we want to modify a commercial product for our specific needs. For example, suppose we are given a language identification system and we want to include a few more languages in the system but we do not have the language dataset used to train the original system. This problem is also pertinent to cases where we want to extend a model, but for computational reasons we do not want to train this model from scratch, even though the whole training dataset is available. In the case where we are only interested in the new classes and we do not care about the old classes and if we have enough training data we can train a new network from scratch. If the new training dataset is small and we can inspect the network parameters, we can apply transfer learning techniques [1] that take the model that was fully-trained for the original categories and retrain it from the existing weights for new classes. In the retraining process we can start from the current set of parameters or even freeze the first layers of the network. Standard transfer learning techniques, however, do not work when we want the network, which is trained only on the new classes to also classify data from the original classes. This issue is related to the notion of catastrophic forgetting, which is the tendency of an artificial neural network to completely and abruptly forget previously learned information upon learning new information [2, 3].

In a wide variety of machine learning problems, the training dataset consists of sensitive data. In such applications, one needs to train a machine learning model without compromising privacy [4]. The learning challenge we address in this study can occur when the data used to train the original system are confidential and we cannot access them in the retraining phase while adding new classes to the system. In recent years privacy preservation in deep learning has became an active research area. The existing literature on deep-learning privacy protection is mostly related to the privacy of the data used for learning a model [5, 6]. The goal is to ensure that the learned model does not reveal any information about the individual entries of the training set. In all previous studies it is assumed that the data are available in the training session and the challenge is to prevent the extraction of the training data from the trained model. Similarily, in our problem, we were given a trained model without its training data, so at the retraining phase we could not access the data with labels that were used to train the original system. In the field of privacy preserving, there are two kinds of machine learning attacks: "black-box" attackers can apply the model to new inputs of their choice, while "white-box" attackers can also inspect the parameters of the model [4]. By considering the adding of new classes to a given system as an "attack" on the original system, in our problem we assume a white-box scenario where we can inspect the model parameters.

Transfer learning uses knowledge from one task to help another. A related problem is incrementally training a single network to learn multiple tasks, where each task evaluated solely on the data from its own dataset (see e.g. [7, 8]). A multi-task situation in our setup occurs when we train the same network to separately classify two disjoint sets of classes. In our class-incremental situation we need to have a classifier that predicts the object class but we do not know whether the object class is in the original or the new set.

In this study we propose a training strategy for adding new classes to an existing network where the dataset that was used to train the network cannot be accessed. As far as we know, previous studies in related learning problems have been restricted to the context of computer vision in various image understanding tasks. Here, however, we apply the proposed method to the task of language identification from the acoustic information conveyed by the speech signal. The applications of language identification systems include multilingual translation systems and emergency call routing, where the response time of a fluent native operator might be critical. In the past few years,

10.21437/Interspeech.2018-1342

with the growing focus on deep neural networks (DNN), DNNs have also been applied to train language classification systems [9][10]. In this study we applied the proposed training procedure to language recognition and report performance on the NIST 2015 Language Recognition i-vector Machine Learning Challenge [11]. In this task there are i-vector examples from 50 languages. We first train a classifier for a subset A of languages and then extend it to classify all 50 languages while only having training examples for the complementary set of languages B. We show that by applying the proposed training strategy, the obtained system can classify all 50 languages.

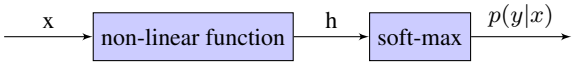
2. Adding Classes to an Existing Classifier

Consider a classifier that was trained to classify an input feature vector x to one of the classes in the set $A = \{1, ..., k\}$. Suppose we want to add m new classes denoted by $B = \{k+1, ..., k+m\}$ to the system and we have training data $x_1, ..., x_n$ with corresponding labels $y_1, ..., y_n$ that all belong to B. We further assume that we can inspect the original model parameters but we do not have access to the training data used to train the original classifier. Our goal is to train a classifier for the set $A \cup B$ using the parameters of the A classifier and the training data labeled by classes from B.

A deep learning classifier consists of two major components. The first is a non-linear transformation $h = h(x)$ and the second is a linear soft-max layer that performs soft classification of a given class-set A:

$$p(y = i|x) = \frac{\exp(w_i^\top h(x))}{\sum_{j\in A} \exp(w_j^\top h(x))}, \qquad i \in A.$$

To simplify notation, we ignore the bias term of the linear input to the soft-max.



To cope with the extended class-set, we apply a transfer learning strategy. We keep all parameters of the original network fixed and only train a new parameter-set $w_{k+1}, ..., w_{k+m}$ which corresponds to the new labels. The modified softmax of the new network is therefore:

$$p(y = i|x) = \frac{\exp(w_i^\top h(x))}{\sum_{j\in A\cup B} \exp(w_j^\top h(x))}, \qquad i \in A \cup B. \quad (1)$$

The challenge is to find a parameters set that works well both for the original classes and the new classes.

Given the training data with labels from B, the objective function we want to optimize is the likelihood function:

$$S(w_{k+1}, ..., w_{k+m}) = \sum_{t=1}^{n} \log p(y_t|x_t) \quad (2)$$

$$= \sum_{t=1}^{n} \log \frac{\exp(w_{y_t}^\top h(x_t))}{\sum_{j\in A\cup B} a_{jt}}.$$

such that $a_{jt} = \exp(w_j^\top h(x_t))$. Note that although we do not have any examples with labels from A, the original class set still appears in the denominator of the soft-max operation. It can be easily verified that the objective function (2) is concave (it is the same function that is optimized in training a logistic regression classifier). Hence, it is easy to obtain the unique maximum parameter set.

The problem with this approach is that the model that optimizes (2) tends to completely forget the original class-set and classify any input to a class from the new set B. We can formally explain this behavior as follows. Let $\mathbf{1}$ be the all-ones vector and α a positive scalar. Given a parameter set $(w_{k+1}, ..., w_{k+m})$ we can form an α-shifted set: $(w_{k+1}+\alpha\mathbf{1}, ..., w_{k+m}+\alpha\mathbf{1})$. The likelihood score (2) of an α-shifted version of a given parameter-set is:

$$S(w_{k+1}+\alpha\mathbf{1}, ..., w_{k+m}+\alpha\mathbf{1}) = \sum_{t=1}^{n} \log \frac{\exp(w_{y_t}^\top h(x_t))}{\sum_{j\in A} \frac{a_{jt}}{b_t} + \sum_{j\in B} a_{jt}}$$

such that $a_{jt} = \exp(w_j^\top h(x_t))$ and $b_t = \exp(\alpha\mathbf{1}^\top h(x_t))$. Assume the activation function used to compute the last hidden layer is either ReLU or Sigmoid, then all the entries of $h(x_t)$ are non-negative and therefore $\alpha\mathbf{1}^\top h(x_t) > 0$ which implies that $b_t > 1$. Each training point $(x_t, y_t \in B)$ thus satisfies:

$$\frac{\exp(w_{y_t}^\top h(x_t))}{\sum_{j\in A} \frac{a_{jt}}{b_t} + \sum_{j\in B} a_{jt}} \geq \frac{\exp(w_{y_t}^\top h(x_t))}{\sum_{j\in A} a_{jt} + \sum_{j\in B} a_{jt}}. \quad (3)$$

Summing (3) over all the training points we obtain that:

$$S(w_{k+1} + \alpha\mathbf{1}, ..., w_{k+m} + \alpha\mathbf{1}) \geq S(w_{k+1}, ..., w_{k+m}).$$

The likelihood score (2) thus monotonically increases as a function of α. The probability to classify an object x to a label $i \in A$ based on the α-shifted parameters is:

$$p(y = i|x) = \frac{\frac{a_i}{b}}{\sum_{j\in A} \frac{a_j}{b} + \sum_{j\in B} a_j} \quad (4)$$

such that $a_i = \exp(w_i^\top h(x))$ and $b = \exp(\alpha\mathbf{1}^\top h(x))$. Eq. (4) implies that for $i \in A$:

$$p(y = i|x) \leq \frac{\frac{a_i}{b}}{\sum_{j\in B} a_j} = c \cdot \exp(-\alpha\mathbf{1}^\top h(x)) \quad (5)$$

such that

$$c = \frac{\exp(w_i^\top h(x))}{\sum_{j\in B} a_j}$$

is a constant that does not depend on α. As α approaches infinity $\exp(-\alpha\mathbf{1}^\top h(x))$ tends to 0 and therefore the probability $p(y = i|x)$ tends to 0. Hence, for each parameter-set of the new labels we can find a better solution that ignores the original set A. This analysis exemplifies the principle of catastrophic forgetting that is caused by the fact that the past data are not available during training.

To overcome the problem described above we regularize the objective function in such a way that we spread part of the mass of each point over the classes from A. The modified objective function is:

$$S(w_{k+1}, ..., w_{k+m}) = \quad (6)$$

$$(1-\epsilon) \sum_{t=1}^{n} \log p(y_t|x_t) + \epsilon \sum_{t=1}^{n} \frac{1}{|A|} \sum_{i\in A} \log p(y = i|x_t)$$

such that ϵ is a parameter that controls the regularization term. In the next section we analyze the optimal value of ϵ as a function of the relative sizes of the old and new class sets. We

Table 1: *Adding Classes Without Original Data (ACWOD) algorithm.*

Input:

- A network that predicts classes in A:

$$p(y=i|x) = \frac{\exp(w_i^\top h(x))}{\sum_{j\in A}\exp(w_j^\top h(x))}, \qquad i \in A.$$

- Training data $x_1, ..., x_n$ with corresponding labels $y_1, ..., y_n$ from a disjoint class-set B.
- No training data with labels from A are available.

Goal: Extend the network to predicts all classes in $A \cup B$:

$$\tilde{p}(y=i|x) = \frac{\exp(w_i^\top h(x))}{\sum_{j\in A\cup B}\exp(w_j^\top h(x))}, \qquad i \in A \cup B.$$

Algorithm: Fix the model parameters and find the parameters $\{w_i | i \in B\}$ that maximize the following concave objective function:

$$S = \sum_{t=1}^{n}\left((1-\epsilon)\log\tilde{p}(y_t|x_t) + \frac{\epsilon}{|A|}\sum_{i\in A}\log\tilde{p}(y=i|x_t)\right)$$

dub this algorithm ACWOD (Adding Classes Without Original Data). The ACWOD algorithm is summarized in Table 1.

An alternative regularization scheme is based on the Knowledge Distillation loss, which Hinton et al. [12] showed to work well to encourage the outputs of one network to approximate the outputs of another:

$$(1-\epsilon)\sum_{t=1}^{n}\log p(y_t|x_t) + \epsilon\sum_{t=1}^{n}\sum_{i\in A}p_{ti}\log p(y=i|x_t)$$

where p_{ti} is the probability that training datum x_t is classified to $i \in A$ in the original network. We found, however, that in our language recognition tasks, a simple flat averaging regularizing (6) achieves similar results. This emphasizes that the main challenge in our learning problem is to avoid the catastrophic forgetting problem and that the reenforcement of the original model is less important.

We note in passing that the analysis presented above is based on the fact that the representation obtained by the ReLU activation function is non-negative. We can replace ReLU by Tanh which can be either positive or negative. Using Tanh without regularization may help a little with the catastrophic forgetting problem, but we found empirically that combining it with the regularization defined above yields inferior results compared to ReLU.

We suggest that ϵ should be a linear function of the proportion of $|A|$ out of the total number of classes. We can then formulate ϵ as follows:

$$\epsilon = c \times \frac{|A|}{|A|+|B|} \tag{7}$$

This means that the weight of the regularization term for each class in the original class set should be constant. The value of c depends on the dataset and on the original model we were given, and was empirically measured to be 0.7 for our problem. In the experiment section we show that setting ϵ to be a linear function of the size of the original class-set is indeed a good approximation of the optimal value.

3. Experiments

3.1. The NIST 2015 dataset

The NIST 2015 language recognition challenge [11] has labeled data from 50 target languages (300 speech segments per language). Table 2 lists the 50 languages arranged by linguistic families. There are 9 singleton languages of two kinds: languages that constitute a family of their own (no known sister languages), such as Japanese, and languages that belong to a broader linguistic family, such as English which is a Germanic language, but were the only representative of this family in our data set. Another interesting unique member of the training set is Creole, which is basically a family of languages. The speech duration of the audio segments used to create the i-vectors for the challenge were sampled from a log-normal distribution with a mean of approximately 35s. The speech segments were derived from conversational telephone and narrow-band broadcast speech data. Each speech segment is represented by an i-vector of 400 components [13]. The NIST challenge also contains an unlabeled dataset that was not used in this study.

Table 2: *The 50 classes of the NIST-2015 language identification dataset arranged by linguistic families.*

Languages	(Sub)Family
Hausa, Somali, Oromo, Arabic, Amharic	Afro-Asiatic
Indonesian, Tagalog	Austronesian
Ukrainian, Polish, Slovak, Czech, Russian, Bosnian	Balto-Slavic
Hindi, Urdu, Punjabi, Bengali	Indo-Aryan
Pashto, Kurdish, Tajik, Farsi, Dari	Iranian
Romanian, French, Portuguese, Spanish	Italic
Shona, Swahili, Zulu	Niger-Congo
Burmese, Cantonese, Mandarin, Tibetian	Sino-Tibetan
Laotian, Thai	Tai-Kadai
Tatar, Turkish, Kyrgyz, Uzbek, Azerbaijani, Kazakh	Turkic
English, Georgian, Greek, Japanese, Khmer, Kosovo, Creole, Armenian, Korean	Singleton

3.2. Implementation details

The classifier we used here is a two fully connected hidden layer Deep Neural-Network (DNN) comprising 200 and 100 neurons each and a soft-max output layer. The activation function was set to be ReLU and the optimization procedure used was mini-batch RMSprop. We also used L_2 regularization to prevent over-fitting. For each language we used 255 speech segments for training and the remaining 45 segments for performance evaluation.

3.3. Results

In the experiments we first trained a DNN classifier to classify a speech segment to one of the classes in a subset A of the 50 languages in the NIST-2015 data-set. Then we used examples from the remaining languages to extend this model so it could classify all 50 languages.

We set ϵ as described in Eq. (7). We computed the performance results as a function of the proportion of the original class-set $|A|$, out of the extended set $|A \cup B|$. We compared our approach to two other alternative methods. The first one was the

original classifier that was trained solely on examples with labels from set A. This method actually maintains the accuracy of set A, while always being wrong when predicting examples with labels from set B. We refer this method as the *original model*. The second naive method is formed by setting ϵ to zero. In other words, this method does not regularize the score function, so the model forgets the original classes, and thus classifies any feature vector as one of the new classes from set B, as explained above. We refer to this method as the *unregularized model*. These two methods are clearly inferior to the proposed approach and they are shown as references to assess the performance of our method.

Fig. 1 shows the language identification performance on the evaluation set as a function of the size of the original class-set. The accuracy of the final model was calculated using the full test set composed of examples with labels from all 50 languages. As shown, we achieved better performance on the entire class-set than the original model even though we did not use any examples from the original class set. When we set $\epsilon = 0$ the catastrophic forgetting principle caused the system to classify all speech segments to a class from the new set. We might have expected that in this model, performance would improved as a function of the size of the new class-set. Fig. 1, however, indicates that if the original class set is small there is a performance degradation. This is due to the fact that in this case the quality of the non-linear representation $h(x)$ that is only learned in the first step is low and the representation does not have enough information to represent all the languages in the new set. By comparison, when training the same model from the beginning with training data that include all 50 languages we achieved an accuracy of 81.2%. Hence, when $|A|$ is a relatively a big proportion of the extended class-set, there is only a small drop in performance.

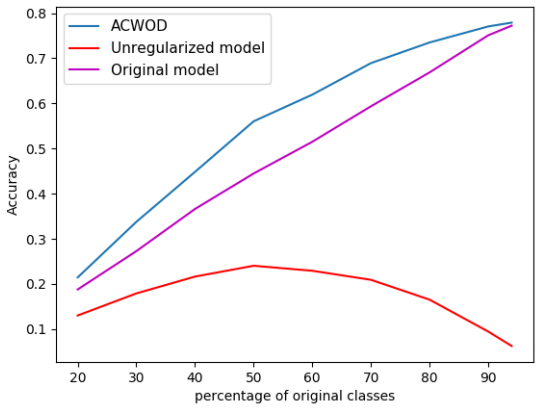


Figure 1: *Language classification accuracy as a function of the percentage of labels in the first (frozen) training stage.*

A major component of the proposed algorithm is the regularization term. Parameter ϵ balances the likelihood and the regularization component of the score. For each $|A|$, we analyzed and found empirically the regularization parameter ϵ that maximized the accuracy of the extended classifier. Fig. 2 depicts the optimal ϵ values as a function of the original class-set $|A|$. It can be seen that optimal ϵ increases monotonically as a function of the original class set. It is also evident that optimal ϵ values can be approximated as a linear function of the proportion of $|A|$ out of the total number of classes, as we suggested in Section 2. We computed the optimal ϵ in other situations and observed similar behavior. The slope of the linear function depends on the dataset and on the original model we were given, but the linear behavior of the importance of the regularization term is a general phenomenon.

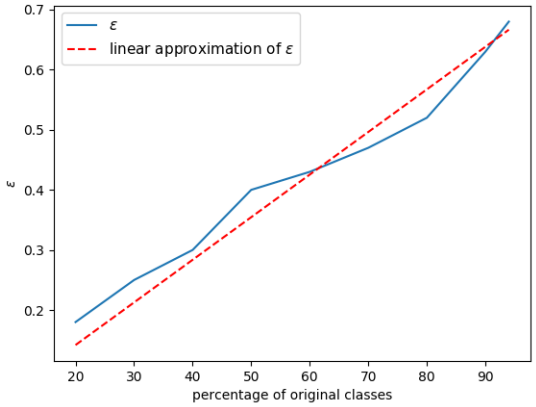


Figure 2: *The value of the optimal regularization weight parameter ϵ as a function of the percentage of labels in the first (frozen) training stages.*

4. Conclusions

In this study we addressed the challenge of getting an existing model to recognize additional classes without any access to the data that were used to train the original classifier. We proposed a deep learning method which enabled us to transfer the learned network to classify the new classes while maintaining its ability to classify the original set. We illustrated the proposed method on the task of language identification and showed that the goal of classifying both the original languages and the additional language can be done with only a relatively small drop in performance when we add a relatively small number of classes, compared to the case of fully observed training data. The proposed method is general and can be applied to any deep learning classification problem.

5. Acknowledgements

This research is partially supported by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Directorate in the Prime Minister's Office.

6. References

[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[2] M. McCloskey and N. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," *The Psychology of Learning and Motivation*, vol. 24, pp. 109–164, 1989.

[3] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[4] M. Abadi, U. Erlingsson, I. Goodfellow, H. McMahan, I. Mironov, N. Papernot, K. Talwar, and L. Zhang, "On the pro-

tection of private information in machine learning systems: Two recent approaches," in *IEEE Computer Security Foundations Symposium*, 2017.

[5] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318.

[6] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," *arXiv preprint arXiv:1610.05755*, 2016.

[7] Z. Li and D. Hoiem, "Learning without forgetting," in *European Conference on Computer Vision (ECCV)*, 2016.

[8] J. Kirkpatrick *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proc. of the National Academy of Sciences of the United States of America (PNAS)*, 2017.

[9] I. Lopez-Moreno, J. Gonzalez-Dominguez, D. M. O. Plchot, J. Gonzalez-Rodriguez, and P. Moreno, "Automatic language identification using deep neural networks," in *ICASSP*, 2014, pp. 5374–5378.

[10] F. Richardson, D. Reynolds, and N. Dehak, "Deep neural network approaches to speaker and language recognition," *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1671–1675, 2015.

[11] "NIST language recognition i-vector machine learning challenge," https://ivectorchallenge.nist.gov/, 2015.

[12] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS Workshop on Deep Learning*, 2014.

[13] N. Dehak, P. A. Torres-Carrasquillo, D. Reynold, and R. Dehak, "Language recognition via Ivectors and dimensionality reduction," in *Interspeech*, 2011.