Duke

TASK-AWARE NEURAL ARCHITECTURE SEARCH

Cat P. Le, Mohammadreza Soltani, Robert Ravier, Vahid Tarokh

2021 IEEE International Conference on Acoustics, Speech and Signal Processing

INTRODUCTION

The design of handcrafted neural networks requires a lot of time and resources. Recent techniques in Neural Architecture Search (NAS) have proven to be competitive or better than traditional handcrafted design, although they require domain knowledge and have generally used limited search spaces. We propose a novel framework for architecture search, utilizing a dictionary of base tasks and the similarity between the target task and the atoms of the dictionary; hence, generating an adaptive search space based on the related base tasks of the dictionary. Lastly, we introduce the Fusion Search (FUSE) algorithm to evaluate and discover the best architecture in the search space without fully training the networks.

METHODS

Given a dictionary of previous task-data pairs. For any target task-data pair, our goal is to find an architecture for achieving high performance on the target task. The proposed Task-aware Neural Architecture Search (TA-NAS) works as follows:

1. **Task Similarity.** Given an incoming task-data set pair, TA-NAS finds the most related task-data set pairs in the dictionary.
2. **Search Space.** TA-NAS defines a suitable search space for the target task-data set pair, based on the related pairs.
3. **Search Algorithm.** TA-NAS searches to discover an optimal architecture in term of performance for the target task-data set pair on the search space.

TASK REPRESENTATION

In our framework, we represent a task-data set pair by neural network. A network architecture is ε-representative of a specific task if it performs sufficiently well on the given task-data set pair. In practice, well-known hand-designed architecture can be chosen as the representation.

TASK SIMILARITY

Let A = (T_A, X_A) and B = (T_B, X_B) be two task-data set pairs, where N_A and N_B are two trained architectures that are ε-representative for A and B, respectively. We can define a dissimilarity measure between A and B as follows:

$$d^{\epsilon}_{A,B} = \min_{N_t \in S_t:\, \mathcal{L}_B(N_t \circ N_A) \geq 1-\epsilon} O(N_t)$$

where S_t is a given transform network search space, and O() is a general measure of complexity (e.g., the number of parameters in a network), and N_t is the network that take the last-layer hidden features of N_A and transform them into N_B's.

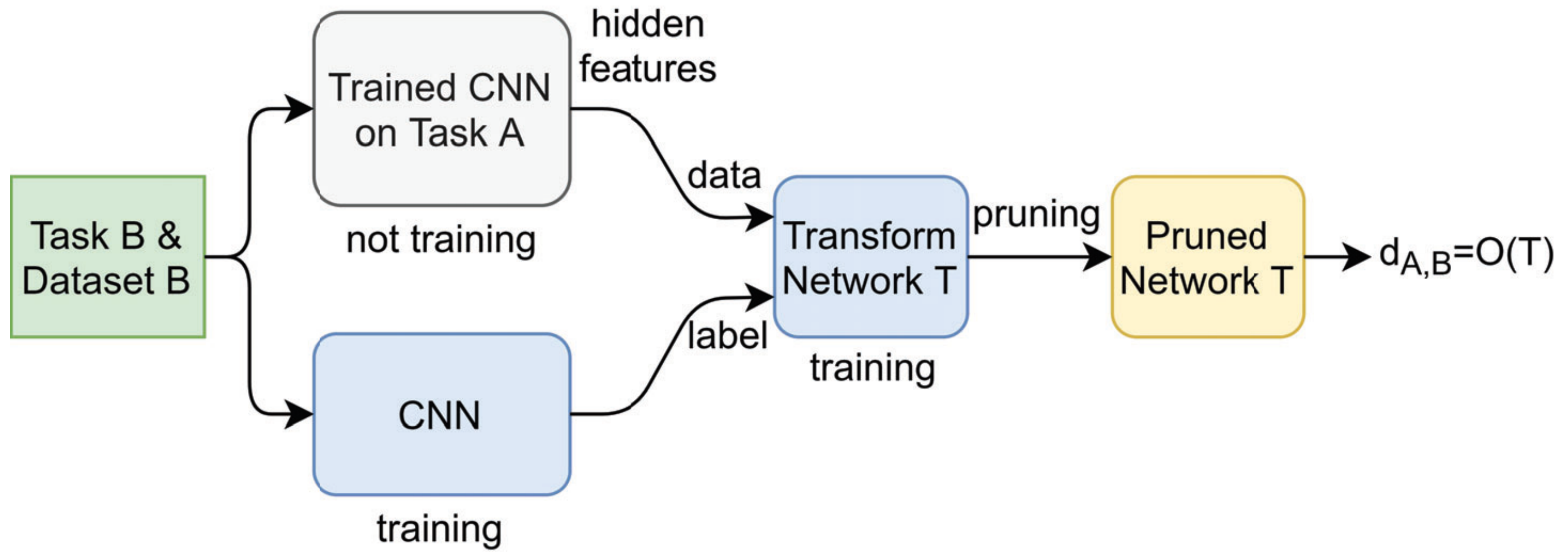
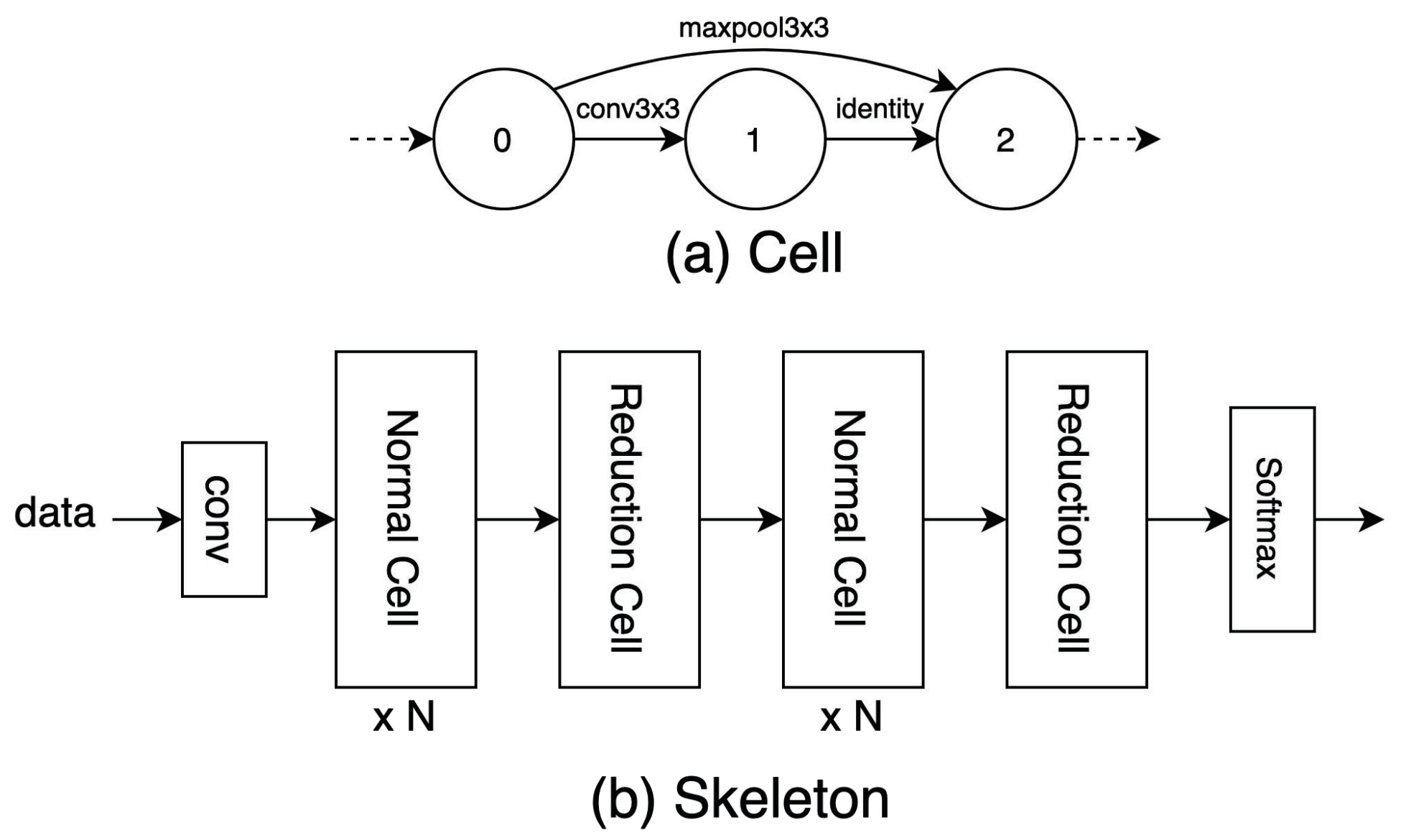


Fig. 1. Illustration of the procedure to compute the distance from task A to task B.

SEARCH SPACE

The search space is defined by the structures of cell and skeleton. A cell is a densely connected directed-acyclic graph of nodes, where all nodes are connected by operations. The skeleton is often predefined. Here, we construct the search space of the target task by combining the skeletons, cells, and operations from only the most similar pairs in the dictionary.



(a) Cell

(b) Skeleton

Fig. 2. Examples of the cell and the skeleton

SEARCH ALGORITHM

Fusion Search (FUSE) is a search algorithm that considers the network candidates as a whole and performs the optimization using gradient descent. For any set of **C** candidates, we relax the outputs by exponential weights:

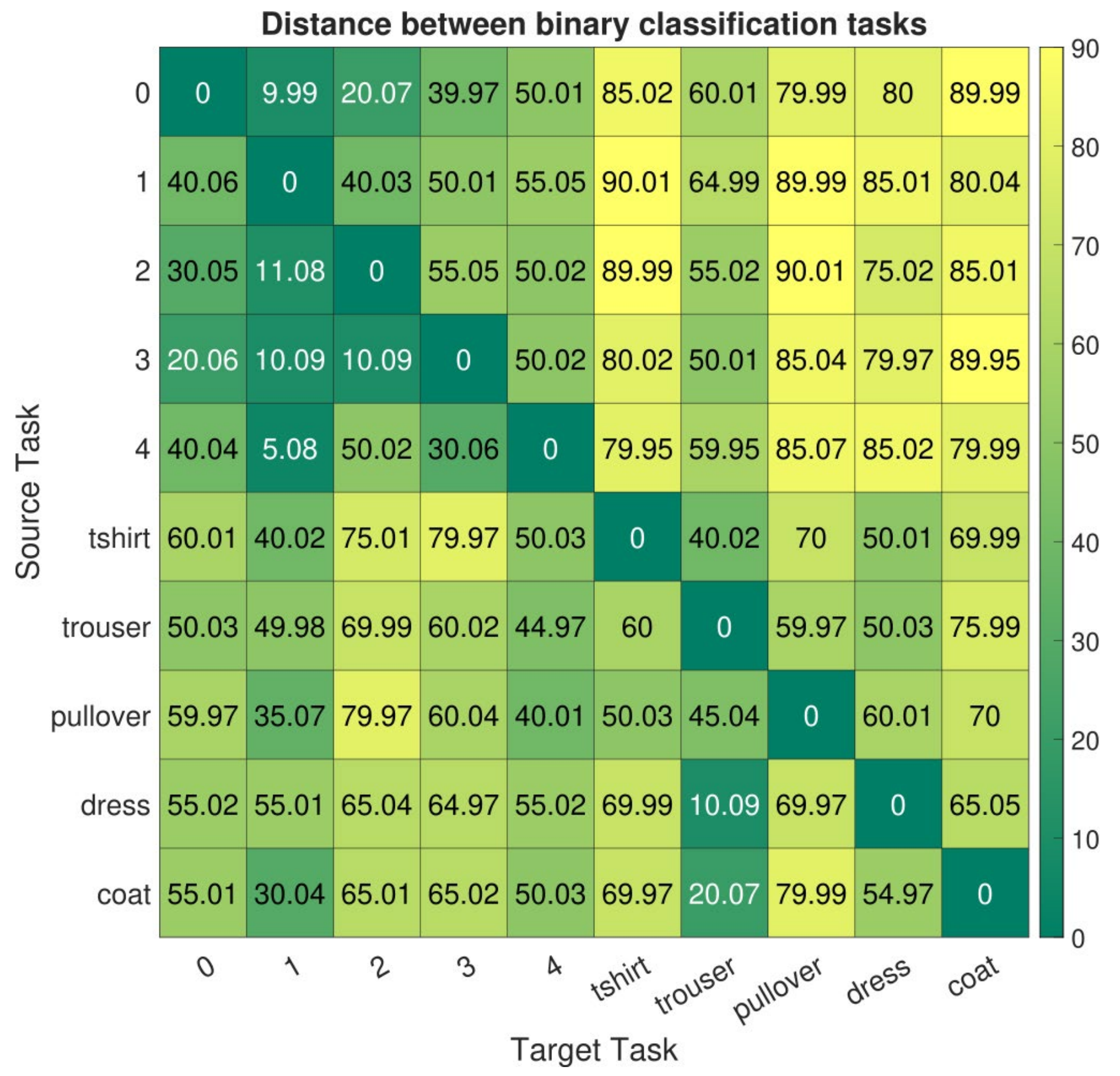
$$\bar{c}(X) = \sum_{c \in \mathcal{C}} \frac{\exp(\alpha_c)}{\sum_{c' \in \mathcal{C}} \exp(\alpha_{c'})} c(X)$$

The training procedure is based on alternative minimization and can be divided into:

- freeze α, train network's weights: $$\min_{w} \mathcal{L}(w; \alpha, \bar{c}, X_{train})$$
- freeze network's weights, update α: $$\min_{w} \mathcal{L}(\alpha; w, \bar{c}, X_{val})$$

RESULTS

For our experiment, we initialize with a set of base binary classification tasks consisting of finding specific digits in MNIST and specific objects in Fashion-MNIST. Let the target task be the binary classification task from Quick, Draw! data set. Tasks from the same data set are more similar than tasks from different data sets.



	0	1	2	3	4	tshirt	trouser	pullover	dress	coat
0	0	9.99	20.07	39.97	50.01	85.02	60.01	79.99	80	89.99
1	40.06	0	40.03	50.01	55.05	90.01	64.99	89.99	85.01	80.04
2	30.05	11.08	0	55.05	50.02	89.99	55.02	90.01	75.02	85.01
3	20.06	10.09	10.09	0	50.02	80.02	50.01	85.04	79.97	89.95
4	40.04	5.08	50.02	30.06	0	79.95	59.95	85.07	85.02	79.99
tshirt	60.01	40.02	75.01	79.97	50.03	0	40.02	70	50.01	69.99
trouser	50.03	49.98	69.99	60.02	44.97	60	0	59.97	50.03	75.99
pullover	59.97	35.07	79.97	60.04	40.01	50.03	45.04	0	60.01	70
dress	55.02	55.01	65.04	64.97	55.02	69.99	10.09	69.97	0	65.05
coat	55.01	30.04	65.01	65.02	50.03	69.97	20.07	79.99	54.97	0

Fig. 3. The distance matrix of base tasks

Architecture	Error (%)	Param (M)	GPU days
ResNet-18	1.42	11.44	-
ResNet-34	1.2	21.54	-
DenseNet-161	1.17	27.6	-
Random Search	1.33	2.55	4
FUSE w. standard space	1.21	2.89	2
FUSE w. task-aware space	1.18	2.72	2

Table 1. The comparison with image classifiers on Quick, Draw!

CONCLUSIONS

We proposed TA-NAS to address the Neural Architecture Search problem. By introducing the task similarity, we can create a restricted search space and quickly evaluate candidates using the FUSE search algorithm. This search algorithm can be applied to find the best way to grow or to compress the current network.