Zero-Shot Task Transfer Learning Using Relational Attention

Jiahao Li[1]

MSc Data Science and Machine Learning

Supervisor:

Luca Franceschi

Massimiliano Pontil

Submission date: 25 September, 2020

Acknowledgements

Firstly, I would like to thank my academic supervisors Dr. Luca Franceschi and Professor Massimiliano Pontil for supervising me to finish this master thesis. I would also like to thanks Dr. Dimitris Stamos for a fruitful discussion during this project.

Secondly, I would like to thank my friend Jiaming Cheng and my brother Jiajie Li. I am appreciated their time in giving feedback for the thesis writing and their support during my undergraduate as well as graduate study.

Lastly, I would love to thank my primary supervisor Dr. Luca Franceschi again for his enthusiasm and patience. His countless support and encouragement help me overcome lots of difficulties in this master project.

Abstract

In zero-shot image classification problems, the cross-modal interaction between images of a class and its corresponding description is to train a model, classifier, that can generalize to unseen classes given its description. The development of zero-shot learning and meta-learning can empower machine learning model to perform zero-shot learning at the task level. The task level zero-shot learning is the generalization of class level learning, which is to provide different models tailored for different unseen tasks. Meta-learning enables the machine learning model to accumulate knowledge from the solved tasks. In this project, we propose a method of solving unseen novel tasks by reusing the knowledge from previous solved tasks given only the unseen task's description.

We address the zero-shot task transfer learning problem from multi-task learning, where the learning model is to learn several tasks at the same time, and meta-learning perspectives. The zero-shot task transfer learning is formulated as a bi-level optimization-based meta-learning problem. In this bi-level optimization problem, the inner objectives is to produce tailored models for different tasks with using the task descriptor, the knowledge from previously solved tasks, and attention mechanism. The outer objective is to update the model's knowledge based on the model's experiences in solving seen tasks. We show that the description can be used to compare the dependency among tasks using attention mechanism and perform zero-shot task transfer learning. The proposed method is evaluated on both synthetic datasets and real-world datasets, and it can produce promising results on zero-shot task learning problem.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we will describe the problem being studied in this dissertation and introduce the motivation behind the problem. We will also discuss our research scope and contributions. Lastly, we will introduce the structure of this dissertation.

1.1 Problem Statement and Motivation

With the fast development of machine learning research and application in recent years, there is an increasing need for using machine learning models to assist people making decisions and solving different kinds of tasks. Also, the booming of the machine learning development toolkit makes the training of the machine learning system cheaper and faster. However, it is difficult for system designers to consider every task the system will encounter in its lifetime. Besides, training of a machine learning system consumes a large amount of labeled data, which is scarce in the real world scenario. Therefore, the machine learning system should be able to exploit the dependency among different kinds of tasks and utilize transferable knowledge from previously solved tasks to solve new tasks.

Even though the labeled data for different tasks is expensive to get in the real world, the acquisition of a task descriptor is relatively easier. Task description is the meta-data of task, which might describe the tasks from the relevant domain knowledge or auxiliary information. For example, the sinusoidal wave can be described by amplitude, frequency and phase shift mathematically. Or the social network profile can reflect new users' interests in recommendation system. Besides, the human can adapt to a new environment or learn a new skill without having experience in the new environment or the new task but a few conceptual knowledge. For instance, if people have some previous experience in cooking, they can learn to cook a new food only by reading the relevant recipes. Therefore, we are going to propose a method which can use the knowledge from previously solved tasks and the new task's description to address the new tasks in this project.

Zero-shot learning is a classical and well-developed research area in image classification for computer vision, in which the semantic description is used to train a classifier to recognize unseen classes. [43, 32, 30, 41, 33] The semantic descriptor can be either a natural language sequence "Light purple petals with orange and black middle green leaves" for describing flowers [32] or a representation vector, such as $\{1, 1, 0, 1, 0\}$ stands for the "dog" class where "dog" is described as flurry, having a tail, being not able to breathe underwater, being carnivorous and not moving slowly in [30].

However, there are only a few works relevant to the generalization of zero-shot learning to the task level specially designed for different areas. Class-level zero-shot learning is to train a model for one specific task. In zero-shot image classification, a trained (multi-class) classifier can be generalized to recognize unseen class given the unseen class' semantic description. Task-level zero-shot learning is to provide different customized models for different tasks with different specifications and/or descriptions. For example, a learning model is trained to generate different models for different kinds of computer vision tasks from the previously learned computer vision tasks. [55, 29] Or the learning model should generate different classifiers to recognize different kinds of animal images given the animals' description. [51, 3]

Another application of zero-shot task transfer learning is the cold-start problem in recommendation system, the cold-start problem is to address the problem that finding out the product the cold users might be interested in or recommend a cold product to users using historical interaction records of warm users and items. The cold users do not have any interaction with any kind of product but some auxiliary information, such as user demographics data and the social network data, they provided during registration. Similarly, the cold products do not have interaction with users/consumers, and only has the product descriptions. Cold users and item recommendation problems are regarded as different zero-shot/few-shot learning tasks in the related works where meta-learning is used to address the cold-start problem. [44, 19, 5, 58]

Multi-task learning models learn several tasks at the same time and allows similar tasks to have similar task parameters, which can improve the efficiency of training and the generalization of models. Inspired by the idea of multi-task learning models from [17, 26, 52], the parameter of different tasks can be learned from a latent representation of task parameter. We propose a method to solve the zero-shot task transfer problem from the multi-task learning and meta-learning perspectives. We assume all task parameters, including seen and unseen tasks, can be inferred from a latent representation of task parameter using task descriptors. The latent representations are regarded as meta-knowledge of the meta-learning model, which is used to maintain knowledge from previous solved tasks.

1.2 Research Scope

This dissertation aims to propose a new method on addressing zero-shot task transfer learning problems in non-linear regression, image classification with proper task definition, and the cold-start recommendation problem from multi-task learning and meta-learning perspectives. We use methods in multi-task learning, meta-learning and attention mechanism to address the zero-shot task transfer learning. The contribution of this work are:

• We train a latent representation of tasks, which encode the knowledge of solved tasks, to predict the task parameter for novel tasks in zero-shot task transfer learning.

A bi-level optimization based meta-learning framework is used to learn a latent representation of task parameters shared among all tasks, including the solved tasks and novel tasks, globally at the outer objective. And the dependency between the novel task and the knowledge of previous solved task is modeled locally to predict task parameters for novel tasks using the latent representation of task parameters at the inner objective.

• We model tasks dependency and reuse the knowledge in the latent representation of tasks to address the unseen tasks with only task description available by using attention mechanism.

Attention mechanism, inspired by the human brain's behavior, is an efficient and interpretable way for assessing the dependency among different entities commonly used in machine learning models. In this project, we deploy the attention mechanism to exploit the dependency between the novel task and the knowledge of previous solved task using their corresponding task descriptions. The results obtained from the attention mechanism is used to retrieve the knowledge from the latent task parameter representation to predict the novel task parameters, which make the zero-shot task transfer learning possible in our proposed method.

We hypothesise that similar tasks would have similar dependency on the latent representation of task. We are going to maintain another latent representation for task descriptors. Inspired from Zaggorukyo & Komodakis [54], the attention transfer method is used to update the latent representation of task descriptors, such that the dependency between task descriptors and the corresponding latent representation can preserve the dependency between task parameters and the latent representation of task parameters.

In this work, we mainly focus on the tasks in the general supervised learning setting, regression and classification, with the available task descriptions. We are not going to study reinforcement learning tasks. Zero-shot task transfer learning for computer vision tasks, such as [29], is also not compatible with our experiment setting because the semantically meaningful task descriptions are not available.

1.3 Structure of the Dissertation

Chapter 2 will introduce relevant background which is helpful to understand the methods used in this project. Chapter 3 is going to describe our proposed methods for zero-shot task transfer learning. Chapter 4 introduces the experimental setup and present the experiment results in both synthetic data and real-world data. Chapter 5 will evaluate the experimental results and give a detailed analysis of the proposed method. Source code is available in the online repository.[1]

[1]https://github.com/Kaho-Lee/Zero-Shot-Task-Transfer-Learning

Chapter 2

Background

This chapter is to introduce the background knowledge relevant to zero-shot task transfer learning. We will firstly introduce the knowledge of zero-shot learning in image classification. Then we will introduce the background knowledge of the attention mechanism, meta-learning and multi-task learning, which are used to constitute the proposed zero-shot task transfer learning model. Lastly, we will give an overview of some relevant papers, which use task descriptors to either improve the performance of learning or perform zero-shot task transfer learning.

2.1 Zero-Shot Learning

In image classification, zero-shot learning is to train a classifier to recognize an unseen class without using visual representation but using the semantic descriptor. Therefore, it is necessary to train an alignment model between class's visual representation and the representation of the semantic descriptor of different classes. There are mainly three divisions for training a zero-shot image classification model, including discriminative approaches, generative approach, and meta-learning approaches. [42] The following sub-sections will introduce several ideas of discriminative approaches and generative approaches. The meta-learning approaches will be introduced at the end of the meta-learning background section.

2.1.1 Discriminative Approaches in Zero-Shot Learning

The discriminatives approaches in zero-shot learning is to either map the visual representation into the semantic representation space [30, 41] or measure the similarity between the visual representation and semantic representation [32, 33, 14] to recognize the novel class images.

Palatucci et al. [30] assume that there are a set of images and the corresponding label data, $\{x, y\}_{1:N}$, and a set of the semantic knowledge base, $\{a, y\}_{1:M}$, where a is the semantic descriptor of a class lying at the p-dimensional semantic space S^p. Then, a semantic output code classifier, $H : X^d \rightarrow Y$, is trained to map d-dimensional image feature to the set of label Y as following

$$\begin{aligned}
&Y = H(X^d) = F(G(X^d)) \\
&G : X^d \rightarrow S^p \\
&F : S^p \rightarrow Y
\end{aligned}$$

where two mapping function, F and G, are trained to map the image feature into semantic space, S^p, and map the semantic representation into label set, Y, respectively. When a new image from unseen class is presented, the classifier will make the prediction based on the encoded semantic representation of image feature generated by G and the ground truth semantic descriptions. Similarly, Socher et al. [41] train a two layers neural network, parameterized by $\Theta \in \{W, M\}$, to map the image feature into semantic space using the following objective during training

$$L(\Theta) = \sum_{y \in Y_s} \sum_{x^i \in X_y} ||a_y - Mf(Wx^{(i)})||^2$$

where $x^{(i)} \in X_y$ is the training image feature, $y \in Y_s$ is the corresponding class label, a_y is the semantic descriptor for class y and f is the non-linear transformation at the first layer of the neural network. Then, when a new image is presented, the image will be classified into either seen classes or unseen classes of the model by using outlier detection methods on the model predictions. For the unseen class (zero-shot case), they assume an isometric Gaussian distribution around each of the novel class's semantic descriptions and make the prediction of the image based on likelihood estimation.

Romera et al. [33] formulate the zero-shot transfer as a minimising linear predictor's loss during training, where the objective is

$$\min_{V \in R^{d \times a}} L(X^T W, Y) + \Omega(W)$$
$$= \min_{V \in R^{d \times a}} L(X^T VS, Y) + \Omega(V)$$

where the matrix factorization approach, $W = VS$ is used to decompose the weights in linear predictor into V and S. V is the matrix mapping the image feature into semantic description space S. Thus, in the zero-shot inference step, the novel class with description $S^{'} \in [0,1]^{d_a \times N}$ can be predicted by using

$$\underset{i}{\text{argmax}}\ x^T VS_i^{'}, i \in (1, ..., z^{'})$$

where d_a is the dimension of the semantic vector and N is the number of novel classes.

Reed et al. [32] formulate the zero-shot learning problem as learning a joint embedding for image feature and semantic descriptor representation. The joint embedding encourages the one specific class's visual representation and the corresponding class semantic representation to be as similar as possible. The zero-shot image classification is performed by measuring the similarity between the visual representation of image features and the semantic representation of the set of unseen class' descriptions. Koulouri and Rostami et al. [14] propose a method for zero-shot image classification by training a joint dictionary learning model. Two dictionaries for image features and semantic descriptors are jointly trained by enforcing the same sparse coding used for reconstructing image features and class semantic descriptors. The image features from unseen classes can be mapped to the domain of semantic descriptor by finding the optimal sparse coding. This method uses the same idea and is proposed by the same author as the relevant work in zero-shot task transfer [34] which we are going discuss in detail in Section 2.5.

2.1.2 Generatives Approaches in Zero-Shot Learning

With the development of autoencoders and Generative Aderversial Network (GAN), generative approaches are also used in zero-shot image classification. In the generative approach, they either use the semantic descriptor to reconstruct images using autoencoders or generate fake images based on the semantic description using GAN. Kodirov et. al. [13] and Li et al. [21] use linear autoencoder to address zero-shot learning problem. For zero-shot image recognition, the encoder part is mapping the visual representation into its corresponding semantic descriptor, then the decoder part can be used to reconstruct the visual representation. The objectives function is as following

$$\min_{W,M} ||X - MWX||_F^2, \ s.t. WX = S$$

where X is the image features, S is the semantic descriptor for the class of X, W is the weight at the encoder part and M is the weights for the decoder. For zero-shot classification, the image from the novel class will first be converted into a semantic descriptor using encoder. Then, the predicted label of images can be obtained by calculating the distance between the learned semantic descriptor and the set of prototype semantic descriptors for different classes.

GAN based methods [20, 49] are also used in zero-shot image classification, where GAN models are to generate fake images of a class based on the semantic descriptors. Although the seen classes and unseen classes are disjoint, the semantic descriptor space are shared by seen classes and unseen classes. Therefore, in the testing step, the images of the unseen class can be generated from the unseen class's descriptor. The recognition of the unseen class can be changed as a conventional supervised learning setting using a multi-class classifier.

2.2 Attention Mechanism

Attention mechanism exists in the human visual system, where human will focus on a part of image instead of the whole image based on some stimulus from the brain. [4] In order to mimic the human brain's behavior, attention mechanisms are proposed for the Computer Vision task, such as image caption [50]. Also, attention mechanisms are used in natural language processing tasks, such as Machine Translation [23, 45] and Question-Answering [53], as well as information retrieval task, such as short text retrieval [31]. The attention mechanism is used to measure the dependency among the tokens in natural language. Attention is also used in transfer learning and knowledge distillation. [54] In short, the attention mechanism is to learn a new representation of input data, feature representation in models' inner layers, or inner variable of machine learning models to improve the training of the learning model. In the following subsections, the formulation of the attention mechanism and the activation function used in the attention mechanism will be introduced.

2.2.1 Attention Formulation

There are mainly two kinds of attention mechanisms, "Hard" attention, and "Soft" attention. Assuming there is a set of d-dimensional representation vectors, r

$$r = \{r_1, ..., r_N\},\ r_i \in \mathbb{R}^D$$

The attention mechanism is used to generate a new contextual representation vector, $Z = \{z_i\}_{t=1}^{N}$, where z_i is the new representation of the i-th entry at the representation vector, r. Therefore, if the input features are used as the input of attention mechanism, the working flow of machine learning model training using attention can be summarised as $X \rightarrow Z \rightarrow Y$. "Hard" attention samples one entry of the representation at a time for model training from a multi-variate Bernoulli distribution. [50]. Let b_t be a set of random variable representing the sampling at t-th time. For example, $b_{t,i}$ is the random variable for r_i at t-th time.

$$p(b_{t,i} = 1 | b_{j<t}, r) = p_{t,i}$$

Thus, the generated new contextual representation of input feature is as

$$z_t = \sum_{i=1}^{N} b_{t,i} r_i$$

However, "Soft" attention uses the weighted average of extracted feature vectors over a probability distribution generated from the softmax function. Softmax function in attention is to model the dependency between a current pointed feature (target) and the other features in the feature vector (source). The soft attention is calculated by

$$z_t = \sum_i C(r_i, r_{j\neq i}) r_i$$

$$C(r_i, r_{j\neq i}) = \frac{e^{score(r_i, r_{j\neq i})}}{\sum_{l=1}^{N} e^{score(r_i, r_{l\neq i})}}$$

Luong et al. [23] and Vaswani et al. [45] propose several ways to model the dependency.

$$score(r_t, r_s) = \begin{cases} r_t^T r_s & \text{dot} \\ \dfrac{r_t^T r_s}{\sqrt{d}} & \text{scaled dot} \\ r_t^T W_a r_s & \text{general} \\ V_a^T \tanh(W_a[r_t; r_s]) & \text{concat} \end{cases}$$

where r can be any kind of representation depending on different model training specifications. d is the dimension of r. r_s represents the source representations, where the model will decide how much attention it will put into it to learn a new representation with respect to the target representation. r_t is the target representation, which is the representation currently pointed by the model. Attention mechanism models the affinity of r_t and r_s in order to determine weights in soft attention. W_a and V_a are both trainable parameters at the attention mechanism.

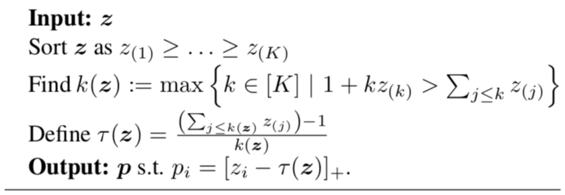
2.2.2 Softmax and Sparsemax

Softmax function is representing a probability distribution over n different classes for multi-class classification or can be used to choose one of several options inside the model [10]. Given a set of representation, $Z = \{z_i\}_{i=1}^K$, softmax function is formatted as

$$\text{softmax}_i(Z) = \frac{e^{z_i}}{\sum_j^K e^{z_j}}$$

Softmax is the most commonly used activation function in the "Soft" attention mechanism to let the model know the importance of each part of input/inner representation with respect to the current state. The softmax results are all larger than zero. However, there are sparsity requirements in some application such as multi-label classification [25]. Or, if the size of the feature vector Z is large, we don't want to see the model put attention into the unimportant features. Martin et al. [25] propose the Sparsemax method, which can produce a sparse posterior distribution but keep the softmax's properties and still being differentiable. The sparsemax operation is shown in Fig. 2.1, where

$[K] := 1, ..., K$ and $[t]_+ := max\{0, t\}$. The basic idea of sparsemax is to set the



```
Input: $\boldsymbol{z}$
Sort $\boldsymbol{z}$ as $z_{(1)} \geq \ldots \geq z_{(K)}$
Find $k(\boldsymbol{z}) := \max \left\{ k \in [K] \mid 1 + kz_{(k)} > \sum_{j \leq k} z_{(j)} \right\}$
Define $\tau(\boldsymbol{z}) = \frac{(\sum_{j \leq k(\boldsymbol{z})} z_{(j)}) - 1}{k(\boldsymbol{z})}$
Output: $\boldsymbol{p}$ s.t. $p_i = [z_i - \tau(\boldsymbol{z})]_+$.
```

Figure 2.1: Sparsemax Algorithm from [25]

probability of $z_{(i)}$, which has a smaller value, to be zero. Martin et al. [25] run experiments on the attention-based machine translation model, where attention using sparsemax activation has slightly better performance than attention using softmax activation.

2.3 Meta-Learning

Meta-learning, which is also understood as learning to learn, is to train a model over several classes/tasks in order to improve the model's generalization. Therefore, the model is able to recognize unseen classes and solve new tasks quicker using fewer training samples. Meta-learning model is trained to learn the generalization knowledge over different classes/tasks instead of data points, which is called meta-knowledge. For example, meta-knowledge can be the initialization strategy of models and the hyper-parameter, such as the learning rate and regularisation parameter. In this section, we will first introduce the format of dataset used in meta-learning. Then, we are going to introduce the meta-learning methods used in few-shot learning. We will review several methods at two sub-divisions of the meta-learning methods, optimization-based methods and metric learning based methods. We will also introduce the application of meta-learning relevant to this project, including the zero-shot image classification and cold-start recommendation problems.

2.3.1 Dataset Format for Meta-Learning

In conventional machine learning model training, the dataset is splted into training set and testing set where they follows the same class/task distribution. However, the dataset in meta-learning is split as meta-training set and meta-testing set, which are disjoint. Meta-training set is used to perform models training as well as update the meta-knowledge. The meta-testing set is used to test the model's performance, but the meta-knowledge is fixed during testing. The meta-knowledge would generate the knowledge, such as model parameter initialization or hyper-parameter, necessary for making predictions on the meta-testing set.

2.3.2 Optimization Based Meta-Learning

Optimization-based meta-learning is trying to learn a representation, which is invariant to the kinds of tasks. Finn et al. [7] propose model agnostic meta-learning (MAML) for learning a good model parameter initialization strategy for few-shot learning. Any new class/task in few-shot learning can be recognized/solved by using a few training samples and gradient descent steps. During model training, MAML is to optimize the performance of $f_{\theta'}$ with respect to model initialization θ, where $f_{\theta'}$ is the task model parameterized by the updated parameters $\theta_i^{'}$ from gradient based optimization. For example, if one step gradient is used to update θ to get $\theta^{'}$, the objective of meta-training is defined as

$$\min_{\theta} \sum_{\tau_i \sim p(\tau)} L_{\tau_i}(f_{\theta_i^{'}}) = \sum_{\tau_i \sim p(\tau)} L_{\tau_i}(f_{\theta - \alpha \nabla_\theta L_{\tau_i}(f_\theta)})$$

where $p(\tau)$ is the distribution of tasks. θ is regarded as the meta-knowledge shared among tasks. The meta-training set is divided as the training set and the validation set, and the training and validation sets have the same task distribution. Gradient updates are performed on the learning model to get $\theta^{'}$ using the training set. Then, the meta-knowledge is updated based on the loss of learning model parameterized by $\theta^{'}$ in the validation set. The tasks can be the different linear/non-linear function in regression, different classes in classification, or sequential decision tasks in reinforcement learning.

Franceschi et al. [9] propose a methods (Hyper-representation) that unifies gradient based hyper-parameter optimization [8] and meta-learning by formulating the meta-learning problem as a bi-level optimization problem. The meta-knowledge is a cross task hyper-representation $h_\lambda : X \rightarrow \mathbb{R}^k$ parameterized by λ. And, a task specific model, $g_\mu^j : \mathbb{R}^k \rightarrow Y^j$ parameterized by μ, is built upon the hyper-representation. Thus, the final learning model is formulated as $g_\mu^j \circ h_\lambda$. During the training of Hyper-representation model, the meta-training set is also divided into the training set and the validation set. The bi-level optimization problem is formulated as following

$$\min_{\lambda \in \Lambda} E(w_\lambda, \lambda, X_{val}, Y_{val})$$
$$w_\lambda = \operatorname*{argmin}_{\mu} L_\lambda(\mu, X_{train}, Y_{train})$$

where $E(w_\lambda, \lambda, X_{val}, Y_{val})$ is the validation loss and $L_\lambda(\mu, X_{train}, Y_{train})$ is the training loss. During training, the inner objective is to find the optimized task-specific parameters conditioned by the current hyper-representation. The hyper-representation is updated by the validation loss of the learning model. When new tasks are presented at testing, only the task-specific models for new tasks are needed to be optimized with a few training samples in few-shot learning.

2.3.3 Metrics Based Meta-Learning

In this section, we will use the image classification problem as examples to describe the metric-based meta-learning. Since training a neural network based classifier needs large amounts of data, metrics-based meta-learning is to learn an embedding for different classes firstly. Then, parametric methods, such as neural network, or non-parametric methods, such as nearest-neighborhood, cosine similarity, or euclidean distance, can be used to recognize the image from a novel class with only a few samples available. [46]

Vinyals et al. [46] propose Matching Network for one-shot learning. In Matching Network, given support set with K classes, $\{x_i, y_i)\}_{i=1}^K$, containing one image for each class, the new query image $\hat{x}$, which has the same class label as one of the instances in the support set, is predicted by

$$P(\hat{y}|\hat{x}, \{x_i, y_i)\}_{i=1}^K) = \sum_{i=1}^{K} C(\hat{x}, x_i) y_i$$

where $C(\hat{x}, x_i)$ is obtained from the softmax distribution over cosine similarity between the embedding for query image and the images in support set.

$$C(\hat{x}, x_i) = \frac{e^{\cos(f(\hat{x}), g(x_i))}}{\sum_{j=1}^{K} e^{\cos(f(\hat{x}), g(x_j))}}$$

where $f(\hat{x})$ and $g(x_i)$ are the encoder for the query image and images in the support set. $\cos(.,.)$ is the cosine similarity measurement.

Snell et al. [40] propose Prototypical Networks for few-shot learning and zero-shot learning. Prototypical Networks learns a prototype for each class, such as using c_k as the prototype for class k, on embedding space. In few-shot learning, the prototype for each class is the mean value of the embedding of its corresponding images generated by the encoder f_θ. And the class distribution for image x can be inferred as

$$p_\theta(y = k|x) = \frac{e^{-\text{dist}(f_\theta(x), c_k)}}{\sum_{k'} e^{-\text{dist}(f_\theta(x), c_{k'})}}$$

where f_θ is the encoder for generating embedding for images and optimized by using negative log-probability, $L(\Phi) = -\text{log}(p_\theta(y = k|x))$. $p_\theta(y = k|x)$ is the probability of image feature x being class k. $\text{dist}(f_\theta(x), c_k)$ is the euclidean distance between the embedding of image x and the prototype for class k. Then, in the testing step, the image in the meta-testing set can be recognized by using softmax over the distance between the image embedding and each prototype in the meta-testing set.

Sung et al. [43] propose Relation Network, which learns both the embedding for images and the parametric metric to assess the relationship between query images and images in the support set. Then, the relation scoring measuring if the query image, x_j, belongs to the same class as a image, x_i, in the support set is as

$$\text{score}_{i,j} = g_\phi(f_\theta(x_i), f_\theta(x_j))$$

where f_θ is the encoder for generating images embedding and g_ϕ is the parametric metrics for generating relation score. The model training is to perform regression on the relation scores onto ground truth labels. The objective is

$$\phi, \theta \leftarrow \underset{\phi,\theta}{\text{argmin}} \sum_{i=1}^{m} \sum_{j=1}^{n} (\text{score}_{i,j} - \mathbf{1}(y_i == y_j))^2$$

where y_i is the ground truth label for query image and y_j is the ground truth label for images in support set. $\mathbf{1}(y_i == y_j))$ equals to 1 when y_i and y_j are in same class, otherwise it equals to 0. In the testing step, the query image and support images in the meta-testing set can be fed into the Relation Network to get the label prediction for the query images based on the relation scores.

2.3.4 Meta-learning in Zero-Shot Learning

The Prototypical Network and Relation Network introduced above have a zero-shot learning counterpart in their proposed methods. In Prototypical Network [40], the prototype of each class is changed as the embedding of its corresponding semantic descriptor. In Relation Network [43], two separate embeddings for semantic description and images are learned. The embedding of semantic description is trained by feed-forward MLP and the embedding of images is produced by a pre-trained DNN model. Then, the embedding for semantic description and images from unseen classes are fed into the parametric method, g_θ, to get the relation score.

2.3.5 Meta-learning in Cold-Start Recommendation

There are mainly three kinds of cold-start problems in the recommendation system, which are 1). recommending warm items to cold users, 2). recommending cold items to warm users and 3). recommending cold items to cold users. "warm" stands for the users/items have both interaction records and auxiliary information, such as product description, user social network data and user profiles, and "cold" stands for the users/items have auxiliary information only. Conventional methods for solving cold-start recommendation problems rely on collaborative filtering, such as [37, 16], and linear content-based feature mapping methods such as [36, 21]. Collaborative filtering (CF) either assess the similarity of the auxiliary information between cold and warm users/items or train a joint matrix factorization model for interaction data and auxiliary information of warm users/items. Then the cold users/items interaction can be predicted from cold users/items' auxiliary information. The content-based model derives a transformation between users/items interaction records and auxiliary information by linear regression or auto-encoder.

But we will not discuss CF and content-based methods in detail in this thesis. However, cold start recommendation is formualted as meta-learning problem where different each user and items are regarded as different tasks. Meta-knowledge is updated and accumulated by performing meta-learning on the warm users/items to learn how to produce tailored recommendation models to different users/items. Cold-start recommendation can be achieved by using the meta-knowledge. In this section, we will give an overview of several meta-learning cold-start recommendation models.

Vartak et al. [44] propose meta-learning methods (LWA) to generate a personalized logistic regression model for cold item recommendation based on the user's historical interaction records. The embedding of users' historical positive and negative interactions are collected and combined with meta-knowledge to produce tailored cold item recommendation. Zhu et al. [58] use a multi-task learning framework, Mixture-of-Expert [38], as meta-knowledge to learn CF representation for users/items' auxiliary information separately, named as Heater. The personalized recommendation is produced by combining the user CF representation and the item CF representation.

MAML [7] is also a popular framework for the cold-start recommendation problem, where the cold-start problem can be addressed by using only a few interaction samples. Lee et al. [19] follow the same idea as MAML to propose a method (MeLu) to learn the parameter initialization strategy for the recommendation model. Since there would be numerous different kinds of users/items and MeLu would lead to local optima using the global sharing parameter only, Dong et al. [5] propose memory-augmented meta-optimization (MAMO) which combines the MAML based methods with memory units designed for users and items. Apart from learning the parameter initialization strategy, MAMO uses attention mechanism to retrieved the entries in memory to generate biased terms added into the parameter initialization for personalized recommendations.

2.4 Multi-task Learning

In this section, we will first go through several multi-task learning methods. Then, we will introduce the recent works of using attention mechanism to address multi-task learning problem.

2.4.1 Multi-task Learning Methods

Multi-task learning (MTL) is to learn several tasks together to achieve different goals, which improves the efficiency of task model training. MTL can also improve the generalization of the model because of the effect of regularisation due to the parameter sharing, feature selection and representation sharing.

Evgeniou & Pontil [6] propose a regularization method on MTL, which assumes tasks in the MTL model are similar. Then, the objectives function, which is reformulated by Franceschi et al. [8], is as following

$$\sum_{t=1}^{T}\sum_{i=1}^{D_{tr}^{t}} l(w_t, x_i) + \sum_{j,k=1}^{T} C_{j,k}||w_j - w_k||_2^2 + \rho \sum_{k=1}^{T} ||w_k||^2$$

where $l(w_t, x_i)$ is the loss function for task t, D_{tr}^{t} is the number of training data for task t, T is the number of tasks in the current MTL model, $C \in R^{T \times T}$ is the task interaction matrix and ρ is a regularisation parameter. $\{w_i\}_{i=1}^{T}$ are model parameters for the tasks in the MTL model. This method encourages similar tasks, evaluated by the task interaction, to have similar parameters when solving different tasks.

Argyriou et al. [1] propose a feature selection method on input data for addressing MTL problem by assuming task parameters, W, are within a low rank space, $w_t \in W$ and $W \in \mathbb{R}^{d \times T}$.

$$\sum_{t=1}^{T}\sum_{i=1}^{D_{tr}^{t}} l(w_t, x_i) + \rho ||W||_{2,1}$$

$$||W||_{2,1} = \sum_{i=1}^{d} \sqrt{\sum_{j=1}^{T} w_{ij}^2}$$

W is a low-rank matrix, such that some entries in W are either zero or very small values and only important feature in input data is used to training the MTL model.

Assuming all task are closely related in MTL is not realistic, and Kumar & Daumé [18] introduce a matrix factorization method, $W = LS$, into MTL, which assumes the task parameters is a flexible grouping of the learned latent representation of task parameter. $L \in \mathbb{R}^{d \times k}$ is the latent representation of task parameter and $S \in \mathbb{R}^{k \times T}$ is sparse vectors for each task in the MTL model. Then, the objectives function is as

$$\sum_{t=1}^{T}\sum_{i=1}^{D_{tr}^{t}} l(w_t, x_i) + \mu ||S||_1 + \lambda ||L||_F^2$$

Kumar & Daumé adopt the alternating procedure to optimize L and S iteratively. With the development of deep learning, Yang et al. [51, 52] introduce Deep Neural Network (DNN) into MTL. Yang et al. [52] uses matrix/tensor factorization to control parameter sharing at each sharing layer of DNN based

MTL model, which is a generalization case of matrix factorization approach similar to Kumar & Daumé [18]. And [51] is to learn a joint embedding of task data and task description, which is going to be introduced at section 2.5.

Deep learning based MTL methods usually learn a shared representation for all tasks in the model, then train multiple task specifics layers built upon the shared layers. Misra et al. [27] propose cross-stitch network to automatically control the sharing of representation for each task. According to the analysis in [27], DNN based MTL model has more sharing in the bottom layers of the model and less or no sharing in the top layer of the model. Thus, Misra et al.[27] use independent DNN models for different tasks and deploy the cross-stitch unit to control the sharing of representation among different tasks adaptively. Assuming there are two tasks, A and B, to be learned, an example of cross-stitch unit at one layer of DNN based MTL model is as following

$$\begin{bmatrix} \tilde{x}_A^{ij} \\ \tilde{x}_B^{ij} \end{bmatrix} = \begin{bmatrix} a_{AA} & a_{AB} \\ a_{BA} & a_{BB} \end{bmatrix} \begin{bmatrix} x_A^{ij} \\ x_B^{ij} \end{bmatrix}$$

x_A and x_B are activation maps from the separated hidden layers of the DNN model for A and DNN model for B in the DNN model. i and j are the position index of the activation map. The cross-stitch take x_A and x_B as input and generates new representation $\tilde{x}_A$ and $\tilde{x}_B$. a_{ij}, $i, j = A, B$, are trainable parameters to control the cross transfer between task A and task B. The new representations are used by the higher layer of the DNN model for task A and task B respectively.

Ma et al. [24] propose a parameter modulation method to model task relationship in MTL, which is called Multi-gate Mixture-of-Experts (MMoE). MMoE is to train n representations with respect to input data independently, then the representations are combined using gating networks and fed into task-specific layers. The formulation of MMoE is as following for each task k

$$\begin{aligned} y_k &= h^k(f^k(x)) \\ f^k(x) &= \sum_{i=1}^{n} g^k(x)_i f_i(x) \\ g^k(x) &= \text{softmax}(W_{gk} x) \end{aligned}$$

where y_k is the prediction for task k, h^k is task-specific layers, and f^k is the combined representation over the n disjoint expert network. $W_{gk} \in R^{n \times d}$ are trainable parameters for n gating networks for task k. Therefore, the similarity among different tasks can be reflected by the softmax distribution of gating network.

2.4.2 Attention in Multi-task Learning

The attention mechanism is also used in MTL to learning relationships among tasks and learning the importance of the features of data or the representations inside the learning model. Zhao et al. [56] use soft self-attention to learn relation among tasks as well as features of training data. They first learn the task-task relationship, using the ground truth label of training data for different tasks and the feature-feature relationship using the features in training data. Then, they learn task-feature dependence relationships through attention alignment between the new task representation generated by task-task attention (target) and the new feature representation generated by feature-feature relation (source) to generate task-specific features as input to task-specific prediction layers in the DNN model. The task-feature dependence model the dependency between current specific tasks and the shared input feature space, which can be regarded as a feature selection strategy. Liu et al. [22] propose Multi-Task Attention Network (MTAN) to learn a completely shared representation and use attention mechanism to select task-specific representation from the shared representation (source) to train multiple computer vision tasks (target).

2.5 Task Descriptor in Task Transfer Learning

In zero-shot learning, each class/task has its own features, label as well as semantic descriptions. The semantic description can either discrete values, continuous values, or text data. Semantic descriptions are meta-data describing a class or task and encode the relevant domain knowledge. Therefore, semantic descriptions are possible to allow the model to reuse the transferable knowledge from solved tasks. The usage of semantic descriptions for zero-shot learning in class-level, such as image classification problems, is explored in lots of zero-shot learning research mentioned in previous sections. However, there is only a few research focusing on using task descriptors in task-level learning. In this section, we will review a few papers relevant to using task descriptions to improve task model training and perform task transfer learning.

Sinapov et al. [39] propose a method on using task descriptions to select a source task at the pool of solved tasks as the initial policy for an incoming novel task in Reinforcement Learning. They measure the advantage, represented by a continuous value scores, of using one solved task as an initial policy over random initial policy for a novel task. Then they train a regression model to predict the advantage score based on the task description difference between new task and solved tasks. The model training is to minimizing the ranking loss to give out a ranking to the solved tasks with respect to the meta-data of a novel task. The solved task with the highest score is selected to be the initial policy of novel tasks. Oh et al. [28] learn parameterized skills for given natural language instructions in a supervised way for multi-task Reinforcement Learning. Then, zero-shot task transfer learning is achieved by making analogies to assess the similarity between different instructions in order to acquire the parameterized skills for executing the new unseen instructions.

Zheng et al. [57] propose a MTL model, which constructs a knowledge graph on the task descriptors. Clustering is performed on the knowledge graph as well as the real training data to generate several groups of tasks to be trained together. The knowledge graph is optimized based on the performance of the MTL model. Therefore, the optimized knowledge graph encodes the relationship and transferability among different tasks.

Isele et al. [34] propose a matrix factorization and sparse coding method (TaDeLL) on both task parameters and task descriptors for zero-shot task transfer learning. TaDeLL first solves each task independently to get the optimized model parameter. If there are T tasks to be learned, TaDeLL assumes that

$$\theta^{(t)} = Ls^{(t)}$$
$$\phi(a^{(t)}) = Ds^{(t)}$$

where $\theta^{(t)}$ is optimized task parameter of the t-th task at the batch of T learning tasks, $a^{(t)}$ is the task descriptor of the t-th task and $\phi(.)$ can be any kinds of linear/non-linear transformation which they make no assumption. L and D are the base of the task parameter and the task descriptor. Both the task parameter and the task descriptor for a task are reconstructed from their corresponding base using the same sparse vector $s^{(t)}$. For the batch of T learning tasks, TaDeLL is optimized by a multi-task learning objective

$$\min_{L,D,S} \frac{1}{T}\sum_{t}^{T}(||\theta^{(t)} - Ls^{(t)}||_{\Gamma^{(t)}}^2 + \rho||\phi(a^{(t)}) - Ds^{(t)}||_2^2 + \mu||s^{(t)}||_1) + \lambda(||L||_F^2 + ||D||_F^2)$$

where ρ is the hyper-parameter to balance the task parameter fit and task descriptor fit. μ and λ are regularisation parameters for the sparse vector and the base, L and D. $\Gamma^{(t)}$ is the hessian of the training loss with respect to the optimized parameter for the t-th learning task. They further decompose the multi-task objective into online updates, such that TaDeLL can be a comparable method in both multi-task learning and life-long learning for zero-shot task transfer learning. The zero-shot task transfer learning is achieved by finding the optimized sparse vector $s^{(t^*)}$ of the equation $\phi(m^{(t^*)}) = Ds^{(t^*)}$ for novel task t^*.

Yang et al. [51] propose a two side network model that unifying the multi-task learning and multi-domain learning by learning embedding for semantic descriptor and data features jointly. We name this model as UMTL in the following chapters, and the architecture of UMTL is shown at Figure 2.2.

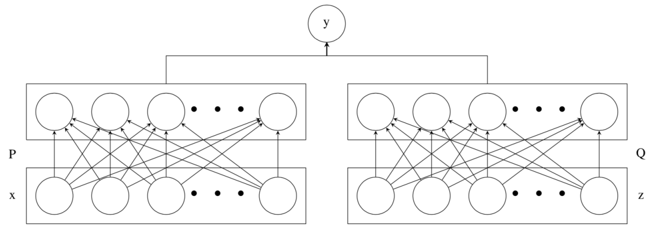


Figure 2.2: UMTL architecture from [51]

The prediction of UMTL model is made by calculating the inner product of embedding of data and semantic descriptor

$$y = (P(x))(Q(z))^T$$

where x is the data features and z is the semantic descriptor. P and Q are the DNN model at the data side and descriptor side and allowed to be defined with any kind of architecture. Thus, this architecture is compatible with zero-shot learning. Chaudhry et al.[3] use the same architecture to perform task-level learning in life-long learning named as A-GEM. The MTL counterpart of A-GEM provides an upper bound of the life-long learning model.

Chapter 3

Methods

This section is to introduce our methods of solving zero-shot task transfer learning from the multi-task learning and meta-learning perspective. We consider performing zero-shot task transfer learning by reusing the knowledge from previously solved tasks following the assumption in [17, 52]. The task parameters, w, are factorized into a latent base, L, and a task-specific component, s, using matrix factorization. We call the latent base of the task parameter as the knowledge base or latent task representation interchangeably, which is learned from previously solved tasks. If the task model is linear with d_1 dimensional inputs and d_2 dimensional outputs, the knowledge base, L, has a size of $\mathbb{R}^{d_1 \times d_2 \times K}$. Following the method in [52], the matrix factorisation is as

$$w = \sum_{i=1}^{K} L_{:,:,i} s_i \tag{3.1}$$

where task specific component, s, is a K by 1 vector and K is the size of knowledge base. Moreover, if the task model is a non-linear multi-layer deep neural network (DNN), the latent base can be constructed by performing matrix factorization at every layer of the DNN model. Although the model is non-linear, the model still follows linear relations at the hidden layers locally before non-linear activation functions. We assume the DNN model used in the project is feed-forward multi-layer perception networks. The convolutional layer and the recurrent model, such as LSTM, are out of the scope of this project.

We will first give an overview of the developed framework and dataset format used in the model. Then, we will explain the bi-level optimization based meta-learning framework and attention mechanisms which we use to model task-task relationships and perform zero-shot task transfer learning.

3.1 Overview

We formulate our framework as a meta-learning problem where the trainable parameter in attention mechanism, W_r, latent task representation, W_{kb}, and latent task descriptor representation, A_{kb}, are meta-knowledge shared globally among different tasks. The attention mechanism is to determine the usage of the entries in knowledge base locally to predict parameters for unseen novel tasks.

In order to prevent the negative transfer, we allow flexible grouping of the entries in latent task representation for solving unseen tasks. Instead of solving a non-convex optimization problem to get the task specific sparse vector, s, like [17, 34], we train a latent representation of task descriptors, and use the attention mechanism to determine the usage of knowledge base based on the real task descriptor and latent task descriptors. The attention map produced from the real task descriptor and latent task descriptor will be trained to preserve the dependency between the real task parameter and latent task representation.

The dataset in this project is split into the knowledge base set, meta-training set, and meta-testing set. The meta-training set have a format of $\{a_i, X_i, Y_i\}_{i=1}^{|T|}$, where $|T|$ is the number of tasks, a_i, $\mathbb{R}^{d_a}$, is the task descriptor for task i, X_i is the data features and Y_i is the corresponding labels. The tasks in knowledge base set are sampled from meta-training set and solved independently to get a set of task parameter, $W_{kb} = [L_1, L_2, ..., L_K]$ and $L_i \in \mathbb{R}^{d \times 1}$, acting as the initialization for latent task representation. d is the dimension of the task parameter and K is the size of the knowledge base. Meta-test set only has a set of task descriptions $\{a_i\}_{i=1}^{|T|}$. The meta-training and the meta-testing set are disjoint.

The initialization of latent task descriptor representation, $A_{kb} = [A_1, A_2, ...\ , A_K]$, is the corresponding task descriptor in knowledge base set and $A_i \in \mathbb{R}^{d_a \times 1}$. As we want the two latent representation can reflect the true dependency between task parameter and task descriptor instead of a random guess, we use the trained task parameters and real task descriptors as initialization parameter.

3.2 Bi-level Optimization and Meta-Learning

The zero-shot task transfer learning is formulated as a bi-level optimization and meta-learning problem as Franceschi et al. [9]. We will first introduce the inner objective, then the outer objective of the meta-learning framework. To make thing simple at present, we assume there is one novel task, t = (a_t, X_t, Y_t), needed to be solved.

Inner Objective. In this project, we hypothesis that the parameter for task model is inferred from the latent task representation, W_{kb}. The goal of inner objective is to get the predicted task parameter, w_t^*, from knowledge base by minimizing the following objectives.

$$w_t^* = \underset{w_t}{argmin}\ L_\lambda(a_t) = \underset{w_t}{argmin} \sum_{i=1}^{K} C(a_t, A_i)||w_t - L_i||_F^2 \tag{3.2}$$

where $C(a_t, A_i)$ describes the contribution of i-th entry in latent task representation on predicting the parameter for task t using the task's descriptor, the latent representation of task descriptor and attention mechanisms. For equation 3.2, there is an analytical solution for the predicted parameter by making the gradient of equation 3.2 with respect to w_t as 0 to get the following expression

$$\begin{aligned} w_t^* &= \sum_{i=1}^{K} \frac{C(a_t, A_i)}{\sum_{i=1}^{K} C(a_t, A_j)} L_i & (3.3) \\ &= C_t W_{kb}^T & (3.4) \end{aligned}$$

where $C_t = (\frac{C(a_t, A_1)}{\sum_{i=1}^{K} C(a_t, A_j)}, ..., \frac{C(a_t, A_K)}{\sum_{i=1}^{K} C(a_t, A_j)}) \in \mathbb{R}^K$. Therefore, the inner objective makes the zero-shot task transfer learning possible using only the inner objective. The predicted parameter for tasks in meta-training set are further used to update the meta-knowledge at the outer objective.

Outer Objective. The goal of outer objective is to optimize the meta-knowledge, including the parameter of attention, W_r, and the knowledge base, W_{kb}, using gradient based optimizer.

$$\underset{W_{kb}, W_r}{min}\ E(w_t^*, X_t, Y_t) + \rho||W_{kb}||_F^2 \tag{3.5}$$

The outer objective is only used at the training steps of the meta-learning model and on the meta-training set. Meta-training loss $E(w^*, x, y)$ can be any kind of loss function depending on the application, such as mean square error for regression and cross-entropy for binary classification. We will discuss the usage of W_r in next section.

This bi-level optimization based meta-learning can be easily extended into the batch setting, where the model can learn more than one task at a time. Assuming there is a batch of tasks, $\{a_t, X_t, Y_t\}_{t=1}^{|B|}$, to be learned during training.

$$\min_{W_{kb}, W_r} \frac{1}{|B|} \sum_{t=1}^{|B|} E(w_t^*, X_t, Y_t) + \rho ||W_{kb}||_F^2 \tag{3.6}$$

$$W^* = CW_{kb}^T \tag{3.7}$$

where $|B|$ is the batch size, C, $\mathbb{R}^{|B| \times K}$, stores the attention vector between the descriptors of the batch of training tasks and latent task descriptor representation. $W^* = [w_1^*, w_2^*, ..., w_{|B|}^*] \in \mathbb{R}^{|B| \times d}$ are the predicted task parameters for the batch of tasks from the inner objective.

3.3 Using Attention to Reuse Knowledge

The attention mechanism can exploit the dependency among different contextual representation and helps the learning model to recognize the important contextual representation during training. We use general soft attention in Luong et al. [23] on task descriptor and latent task descriptor representation to determine the the importance of entries in latent task representation for learning the novel tasks. The analytical solution of the inner objective is to generate task parameters for novel tasks using the useful content in the knowledge base extracted by the attention mechanism. The importance of the i-th entry in latent task representation for learning a new task t is modeled using Softmax activation

$$C(a_t, A_i) = \frac{e^{a_t^T W_r A_i}}{\sum_{j=1}^K e^{a_t^T W_r A_j}} \tag{3.8}$$

where W_r, $\mathbb{R}^{d_a \times d_a}$, is trainable parameter in attention mechanism and optimized at the outer objective of bi-level optimization problem. Different forms of attention can also be used.

It is unrealistic to assume all tasks are similar to each other and necessary to control the parameter sharing when learning multiple tasks. Thus, we also use sparse activation function, Sparsemax, in the attention mechanism, such that the estimated task parameter for novel tasks is related to a few but not all latent tasks. Only similar tasks are allowed to share one or more entries from the knowledge base in common. The advantage of using Sparsemax and Softmax is that they can learn the grouping of tasks adaptively from task descriptors for meta-learning model training. We will compare the performance between using Softmax as the activation function and using Sparsemax as the activation function in the experiment section.

3.4 Attention Transfer

We hypothesis that similar tasks would have similar dependency on the latent task representation. We let the attention vector between tasks' descriptors and the latent representation of task descriptor to preserve the dependency between the parameter of tasks and the latent task representation. The scaled-dot product attention in Vaswani et al. [45] is used to model the dependency because of its efficient implementation and numerical stability. Different forms of attention are also open for discussion. The dependency within latent tasks representations, $P_1 \in \mathbb{R}^{K \times K}$, is modeled by

$$P_1 = \text{softmax}(\frac{W_{kb}^T W_{kb}}{\sqrt{d}}) \tag{3.9}$$

The dependency, P_2, between the tasks in meta-training set and latent tasks representation in the knowledge base is

$$P_2 = \text{softmax}(\frac{W_{train}^{*^T} W_{kb}}{\sqrt{d}}) \tag{3.10}$$

where $P_2 \in \mathbb{R}^{|B| \times K}$ and W_{train}^* is the predicted task parameters from the inner objective. Similarly, the dependency within latent task descriptor representation and the dependency between task descriptor and latent task descriptor representation are

$$Q_1 = \text{softmax}(\frac{A_{kb}^T A_{kb}}{\sqrt{d_m}}) \tag{3.11}$$

$$Q_2 = \text{softmax}(\frac{A_{train}^T A_{kb}}{\sqrt{d_m}}) \tag{3.12}$$

The latent representation of the task descriptor, A_{kb}, is updated by using the attention transfer method from Zagoruyko & Komodakis [54]. The attention transfer is to enforce the attention map produced by task descriptors and A_{kb} to be as similar as the attention map produced by predicted task parameters and latent task representation, W_{kb}. The objective of updating A_{kb} is

$$\underset{A_{kb}}{min} \frac{1}{|K|} \sum_{i=1}^{K} \sum_{j=1}^{K} (P_1^{ij} - Q_1^{ij})^2 + \frac{1}{|B|} \sum_{i=1}^{|B|} \sum_{j=1}^{K} (P_2^{ij} - Q_2^{ij})^2 + \mu ||A_{kb}||_F^2 \tag{3.13}$$

where i and j are the indexes of the attention map. The attention transfer is not been minimized at every update of outer objectives. Both the outer objectives and attention transfer are taken one-step gradient update together for updating meta-knowledge efficiently.

In case of any ambiguity, the attention in section 3.3 is to determine the usage of each entry in knowledge base to predict the parameter for different tasks, and the attention discussed in this section is to preserve the dependency. The pseudo-code of the zero-shot task transfer learning model's training and prediction are listed in Algorithm 1 and Algorithm 2.

Algorithm 1: Zero-Shot Task Transfer Learning Training

```
   Input:
   K - number of latent tasks; $|B|$ - batch size of training;
   $\rho$, $\mu$ - regularisation parameter for $W_{kb}$; and $A_{kb}$
   $\eta_1$, $\eta_2$ - learning rate for outer objectives and attention transfer
   Output: $W_{kb}$, $A_{kb}$, $W_r$
   Data:
   Meta-Training set: $T_{train} = \{a_i, X_i, Y_i\}_{i=1}^{|T|}$
1  Sample K tasks $T_{kb} = \{a_{kb}^i, X_{kb}^i, Y_{kb}^i\}_{i=1}^{K} \sim \mathrm{T}_{train}$
2  for k=1 to K do
3      $\mathrm{W}_{kb}(:,k) \leftarrow \underset{w_k}{\operatorname{argmin}}\, L(X_{kb}^k, w_{kb}^k, Y_{kb}^k)$
4      $\mathrm{A}_{kb}(:,k) \leftarrow a_{kb}^k$
5  while not converge do
6      Sample $|B|$ tasks $\{a_i, X_i, Y_i\}_{i=1}^{|B|} \sim \mathrm{T}_{train}$
       /* Inner Objectives                                              */
7      for $i$ = 1 to $|B|$ do
8          $[\mathrm{C}(\mathrm{a}_i, A_1), C(a_i, A_2), ..., C(a_i, A_k)] \leftarrow Eq.(3.7)$
9          $\mathrm{w}_i^* \leftarrow argmin_{w_i} \sum_{j=1}^{K} C(a_i, A_j)||w_i - L_j||_F^2$
       /* Outer Objectives                                              */
10     $W_{kb} \leftarrow W_{kb} - \eta_1 \nabla_{W_{kb}} Eq.(3.6)$
11     $W_r \leftarrow W_r - \eta_1 \nabla_{W_r} Eq.(3.6)$
12     $A_{kb} \leftarrow A_{kb} - \eta_2 \nabla_{A_{kb}} Eq.(3.13)$
13 return $W_{kb}, A_{kb}, W_r$
```

Algorithm 2: Zero-Shot Task Transfer Learning Prediction

```
   Input: $W_{kb}, A_{kb}, W_r$, Task Descriptor of a Novel Task $\mathrm{a}_i^{new}$
   Output: Model for the Novel Task $\mathrm{f}_{w_i^{new}}(*)$
1  $[C(a_i^{new}, A_1), C(a_i^{new}, A_2), ..., C(a_i^{new}, A_k)] \leftarrow Eq.(3.7)$
2  $\mathrm{w}_i^{new} \leftarrow argmin_{w_i} \sum_{j=1}^{K} C(a_i^{new}, A_j)||w_i - L_j||_F^2$
3  return $f_{w_i^{new}}(*)$
```

Chapter 4

Experiment

We perform empirical evaluations on two synthetic datasets firstly to test the effectiveness of the proposed method on zero-shot task transfer learning (**ZSTL**) in both linear and non-linear models. Then ZSTL is tested on two image dataset, **CUB** and **AwA2**, for zero-shot image classification by defining "task" properly and one recommendation dataset **LastFM** for cold-start user recommendation. Both Softmax and Sparsemax [25] are used as the activation function in attention mechanism, which is denoted as ZSTL(Softmax) and ZSTL(Sparsemax).

Most of zero-shot task transfer learning model are designed for a specific domain, such as [39, 28] for reinforcement learning, [29] for computer visions and [19, 5, 44, 58] for cold-start recommendation. We consider making the model compatible with the general supervised learning setting, regression, and classification. We also choose two models, **TaDeLL** and **UMTL**, which can be applied to the general supervised learning settings as well.

TaDeLL [34] trains a coupled dictionary for the task parameter and the task descriptor. The reconstruction of the task parameter and task descriptor of a task from the base of the task parameter and task descriptor shares the same sparse vector. However, it is not efficient for the machine learning model with a large number of parameters, such as DNN. Also, it makes a strong assumption that the coupled dictionary sharing one single sparse vector, which is not feasible in the real-world scenario. Therefore, we only report selected experiment results for TaDeLL and would explain why it cannot fit into the experiment setting.

UMTL [51, 3] is the two-sided neural network for learning the embedding of the task data and semantic descriptor jointly, and the embedding of data and descriptor are combined when making predictions. This model allows to define the architecture freely at both sides [51]. UMTL is compatible with all the experiments of this project. If the linear model is used as the task model, the suggested architecture, which includes one linear layer at each side of the network, in [51] is used. However, if the task is implemented using the deep neural network, the architecture follows the design in A-GEM[3] where there is the same deep neural network as the task model at the data side and one linear layer in the semantic descriptor side.

LoCo[36] is specially designed for cold-start user recommendation. Since we assess ZSTL performance on the cold-start recommendation problem, it is necessary to compare with a cold-start recommendation model to check if ZSTL can produce promising results. LoCo is to learn a low-rank linear regression model, which directly transforms the auxiliary information of users into the user-item interaction records. LoCo is efficient in implementation and training, and can also achieve comparable performance as DNN based models. The meta-learning based methods we introduce in Section 2.2.5 are not compatible with our experiment setting. LWA [44] is not compatible with the cold user problem. There is not any pre-trained CF model necessary for Heater [58]. Although MeLu [19] and MAMO [5] are compatible with cold-start users recommendation, they focus on few-shot learning. However, ZSTL focus on the zero-shot learning paradigm.

Hyper-parameters We will set a fixed learning rate as $5e-4$ for outer objectives update and attention transfer, and Adam optimizer [12] is used. Besides, we also fixed the size of the latent base of the task parameter as $K = 10$, empirically. The effect of K will be discussed in the next chapter. Then the hyper-parameters to be tuned in the ZSTL are the regularisation parameters, ρ and μ, for W_{kb} and A_{kb}. The regularisation parameter ρ and μ are chosen independently over the set $\{1, 1e-1, 1e-2, 1e-3, 1e-4, 1e-5\}$. 20% of training tasks are randomly chosen as validation data. The best combination of regularisation parameters is chosen based on the validation loss. We use mean square error (mse) for regression, accuracy for classification, and mean average precision (mAP) for the recommendation problem as hyper-parameters selection metrics.

Evaluation Metrics We use Y as the ground truth label and $\hat{Y}$ as the predicted label from the learning model. For the regression problem, mean square error (mse) is used as the evaluation metric.

$$\text{mse} = \frac{1}{N}(Y_i - \hat{Y}_i)^2$$

For classification and recommendation problem, the model predictions are classified as true positive (TP), true negative (TN), false positive (FP) and false negative (FN). Accuracy is used metric for binary classification and calculated by

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}}$$

Precision (P), recall (R) and mean average precision (mAP) are used as evaluation metrics for recommendation problems.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}$$
$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}$$

Average precision precision consider the ranking of predictions and calculated as

$$\text{Average Precision} = \frac{\sum_{k=1}^{N}(\text{P}(k) \times \text{Rel}(k))}{\text{number of relevant data}}$$

k indicates the ranking cut off at position k predicted by the model. Rel(k) equal 1 if the ground truth relevancy of the k-th predicted most relevant data is relevant. Otherwise, Rel(k) equal 0. P(k) is the precision of the recommendation model for the k most relevant predictions. mAP is the mean value of average precision for all tasks.

The synthetic data generation, data pre-processing, sampling strategy for tasks' dataset as well as the experiment results are in the following sections.

4.1 Synthetic Data

We firstly re-implement the synthetic binary classification experiment in [34] to verify the effectiveness of ZSTL in the linear task model. Then generating synthetic data on sinusoidal wave regression for assessing the effectiveness of ZSTL in a non-linear DNN based task model.

	Synthetic Binary Classification	Synthetic sine wave
TaDeLL	0.847 $\pm$0.025	N/A
UMTL	0.893$\pm$0.007	0.0097$\pm$0.0023
ZSTL(Softemax)	**0.894 $\pm$ 0.01**	**0.0053 $\pm$ 0.004**
ZSTL(Sparsemax)	0.893 $\pm$0.01	0.0064 $\pm$0.001

Table 4.1: Experiment results of synthetic data; **Left** binary classification accuracy with 95% confidence interval; **Right** mean square error of sine wave regression with 95% confidence interval

Synthetic Binary Classification: Following the design of TaDeLL [34], base for task parameter, L, and task descriptor, D, are firstly generated independently from standard normal distribution, $L \in \mathbb{R}^{d\times k}$, $D \in \mathbb{R}^{d_a\times k}$. d is the dimension of task parameter, d_a is the dimension of task attribute and k is the size of the dictionaries. $s^{(t)}$, $\mathbb{R}^k$, is the sparse vectors for t-th generated binary classification task with sparsity level 0.5. $s^{(t)}$ is generated from a random uniform distribution between 0 and 2, and any entry which is larger than 1 is changed to 0. The task parameter is generated by $w^{(t)} = Ls^{(t)}$ and task attribute is generated by $a^{(t)} = Ds^{(t)}$. In this experiment, every task shares the same input data, $X \in \mathbb{R}^{d\times n}$, where the first element in the features is 1 and the other features are drawn from the standard normal distribution. The corresponding label of input data for a specific task is generated by $y^{(t)} = \text{sigmoid}(w^{(t)T}X)$. Besides, n is the number of data points in X. In this experiment, d=8, d_a=5, n=20 and k=6. There 100 tasks generated as training tasks and 100 tasks generated as zero-shot testing tasks. Besides, for ZSTL, 10 tasks are randomly selected from training tasks as the initialization of the knowledge base and not used in model training as well as model testing.

Synthetic Sinusoidal Wave: Task descriptor for sinusoidal wave are the amplitude, frequency and phase shift. In generation of synthetic sine wave, frequency is kept as 1, amplitude, A, is equally sampled from the interval $[0.4, 2.0]$ with an offset 0.2 and phase shift, ψ, is equally sampled from the interval $[0, 2\pi]$ with an interval $\frac{\pi}{8}$. Thus, there are 144 tasks in total being generated. 200 data points, X, are randomly sampled from the interval $[-2\pi, 2\pi]$. Then, the corresponding label, Y, are calculated by $y = A * \sin(x + \psi)$. 100 tasks are used as forming the meta-training set and 44 tasks are used to form the meta-testing set. Similarly, 10 tasks from meta-training set are for the initialization of knowledge base and not show up in model training and model testing.

For **ZSTL**, we use linear logistic regression as the task model for synthetic classification experiment and the DNN model with 2 hidden layers with 40 units and relu activation as the task model for sine wave regression. For binary synthetic classification, the architecture of **UMTL** include one linear layer with 10 outputs is used at each side of the model. For synthetic sine wave regression, since architecture in A-GEM is not suitable for regression problem and UMTL allows to define the DNN model with great flexibility, two hidden layers with 40 units and relu activation are used in both sides for **UMTL**. Linear task model is also used in **TaDeLL** for synthetic classification. The experiment results of **TaDeLL** in sine wave regression are not reported because **TaDeLL** is numerically unstable and dose not produce meaningful results. Also, **TaDeLL** involves a matrix inverse of the Kronecker product between the task parameter and the base, which is computationally costly. Therefore, TaDeLL is not applicable to DNN-based task models. The experiment results on these two synthetic dataset are reported in Table 4.1.

4.2 Zero-Shot Learning

CUB-200-2011 [47] is a fine-grained image dataset containing 11788 images belonging to 200 different kinds of birds. Each class in the dataset has the varying number of images ranging from 41 to 60. There are two set of semantic description vectors, binary values attributes and continuous values attributes, provided in CUB. We use continuous value attributes for the experiment. In order to prevent infinity values in exponential function when using the attention mechanism, the task attributes are normalized by divided all attribute values by 100. Besides, ImageNet-pretrained ResNet101 [11] with replacing the last linear output layer is used to extract the feature vector of each image. Following the multi-task learning experiment in Yang & Hospedales [51], the CUB dataset is decomposed into 200 different one-vs-rest binary classification tasks, and the attribute of a class of bird is used as the task descriptor. In order to make the model not biased to any specific task during training, 40 images belonging to the corresponding class and 40 images from the whole dataset are randomly sampled to form the dataset of a task. 140 tasks are randomly selected into meta-training set and 50 tasks are randomly selected into meta-testing set. 10 tasks are randomly selected as knowledge base initialization.

AwA2 [48] is another fine-grained image dataset. AwA2 contains 37,322 images from 50 classes of animals with both binary and continuous attributes as well. Binary attributes are used in this experiment. In AwA2, each class contains an average of 746 images, but the least populated class mole has only 100 images [48]. Similarly, ILSVRC-pretrained ResNet101 with replacing the last linear output layer is used to extract the image features. The data sampling for each task is the same as the sampling process used in **CUB**, where 100 images from the corresponding class and 100 images from the whole dataset are sampled. 40 tasks are randomly selected into meta-training set and 10 tasks are randomly selected into meta-testing set. 10 tasks from meta-training set are randomly selected as knowledge base initialization.

Both **ZSTL** and **TaDeLL** also use a linear logistic regression as their task model. The implementation details of **UMTL** are using one linear layer with 10 outputs for each side. The experiment results are reported as binary accuracy and shown in Table 4.2.

	CUB	AwA2
TaDeLL	0.669±0.022	0.538 ±0.0301
UMTL	**0.717 ± 0.012**	0.625 ±0.015
ZSTL(Softemax)	0.693 ±0.012	0.611 ±0.036
ZSTL(Sparsemax)	0.708 ±0.012	**0.63 ± 0.0589**

Table 4.2: Experiment results on zero-shot image classification tasks. Results are reported as classification accuracy with 95% confidence interval.

4.3 Cold-Start Recommendation

LastFM [2] is a recommendation system dataset with auxiliary social network data. There are 1892 users, 17632 artists, and 186479 tags made by users for artists. In this experiment, we will address the cold-start user recommendation problem by training of a personalized model for each user as a binary classification task. The personalized model will decide whether to recommend an artist to a new user. The binary user-user friendship interactions are auxiliary information (task descriptor) for users in this experiment. Besides, the tags of each artist given by warm users are used as the features of each artists. Artists with tag information are data features and the binary user-artist listening interactions are labels for the different tasks in this experiment.

80% of users are selected as the meta-training set (warm users) and 20% of users are meta-testing set (cold users). There are average values of 49.067

	@20			**@50**			**@100**		
	mAP	P	R	mAP	P	R	mAP	P	R
LoCo	0.5265	0.3377	0.1356	0.4395	0.2313	0.2354	0.3796	0.1572	0.3192
UMTL-LR	0.4608	0.2814	0.1138	0.3816	0.983	0.2	0.3258	0.1409	0.2839
UMTL-NN	0.4763	0.2838	0.1147	0.3873	0.2093	0.2095	0.3291	0.1516	0.2985
ZSTL(Softmax)-LR	0.4025	0.2404	0.09671	0.3328	0.1718	0.1728	0.2851	0.1225	0.2469
ZSTL(Sparsemax)-LR	0.4518	0.2772	0.1121	0.3725	0.1934	0.1949	0.3185	0.1379	0.2780
ZSTL(Softmax)-NN	0.3868	0.2363	0.09511	0.3267	0.1687	0.1698	0.281	0.1187	0.2393
ZSTL(Sparsemax)-NN	0.4474	0.286	0.1159	0.3733	0.1955	0.1977	0.3159	0.1392	0.2815

Table 4.3: Experiment results on cold start user recommendation in **LastFM** dataset

binary user-artist listening interactions for users, with a maximum value of 50 and a minimum value of 1. For the meta-training set, in order to address the sparsity and imbalance problem in user-artists interactions, 100 interactions are randomly sampled for each user to form the tasks dataset, which include all of his/her listening interactions and randomly sampled non-listening interactions. 10 tasks from the meta-training set are randomly selected as the initialization of the knowledge base. For the meta-testing set, all of the listening and non-listening records are used as task dataset.

Both linear and DNN models are used as task models in this experiment. **ZSTL** uses the linear logistic regression model (ZSTL-LR) and DNN model with one hidden layer with 200 units and relu activation (ZSTL-NN) as its task model. For linear model implementation of **UMTL**, one linear layer with 10 outputs are used at each size (UMTL-LR). The DNN based **UMTL** (UMTL-NN) is same as A-GEM [3], where one 200 units hidden linear layer and relu activation are used at the data side. One linear layer with 200 units is used at the descriptor side. There is not enough memory to support the computation of **TaDeLL**, thus results are not reported. A cold-start recommendation method **LoCo** is used as the comparison method to check if **ZSTL** can generate meaningful results. mAP, precision and recall at top 20, 50 and 100 ranked items are used as evaluation metrics. The results are in Table 4.3.

Chapter 5

Discussion and Conclusion

In this chapter, we will firstly review the experiment results. Then we will give a discussion on the design choices of ZSTL including the effectiveness of the size of the knowledge base, the difference between using Softmax and Sparsemax in the attention mechanism, and visualize the effectiveness of the proposed methods. Lastly, we will summarise and give a conclusion on the whole project and discuss the potential future research direction on improving zero-shot task transfer learning.

5.1 Analysis of Results

Table 4.1, 4.2 and 4.3 show all the experimental results. ZSTL can outperform matrix factorization based method TaDeLL in all experiments. Because ZSTL relaxes the constraint in TaDeLL that both task parameters and tasks descriptor follow the same linear relationship with their corresponding base. ZSTL learns the relationship between task and latent task representation adaptively from the task descriptor. The results from the attention mechanism will make a flexible grouping of the latent task representation to perform training on the seen tasks and predict task parameters for novel tasks.

However, ZSTL can only achieve comparable results with the results from DNN based model, UMTL, when the number of task descriptions is limited or task descriptors contain too much information in a limited dimensional space, such as 3-dimensional descriptor for sine wave. If the task descriptors contain more details, ZSTL performs worse than UMTL. Since UMTL allows a flexible definition of the network architecture, the performance will be varying with different architecture and we follow the suggestion in UMTL paper [51] and A-GEM [3]. According to the experiment results, ZSTL achieves comparable results with UMTL in the experiments of synthetic sine regression and AwA2. However, ZSTL underperforms UMTL in the experiments for CUB and LastFM. The statistic of the number of tasks and the dimension of the task description vector in different dataset is in Table 5.1. UMTL has clear advantages over ZSTL in those experiments with more tasks and more details in the task description. For the cold-start recommendation experiment, even though ZSTL can produce promising results, LoCo outperforms both our ZSTL model and the UMTL model.

	Sine	AwA	CUB 2	LastFM
Number of Tasks	144	50	200	1892
Dimension of Task Description	3	85	312	1892

Table 5.1: Statistic of the number of task and dimension of THE task descriptor in different datasets

Learning embedding jointly for data and semantic descriptions has already been shown to be successful in improving the performance in multi-task learning [51] and zero-shot task transfer learning and life-long learning [3]. However, ZSTL is to predict the parameter for the task model which takes data as input only. The future improvement on ZSTL could consider the task model which can learn from semantic descriptors as well.

We notice that there is a very recent work in using meta-learning and attention mechanisms to tackle the cold-start recommendation proposed by Dong et al. [5] and named as MAMO. MAMO learns the parameter initialization strategy for the task model which takes both data features and semantic descriptions as input. However, according to the results in Dong et al.[5], MAMO dose not show clear advantage over their comparison methods. Besides, there are two difference between our proposed methods and MAMO. Firstly, we design our model to be compatible with genernal supervised learning task, including non-linear regression, zero-shot image classification as well as the cold start recommendation problem. Secondly, we aim at performing task transfer learning in a zero-shot way. But MAMO instead is specially designed for cold-start recommendation and focusing on few-shot learning.

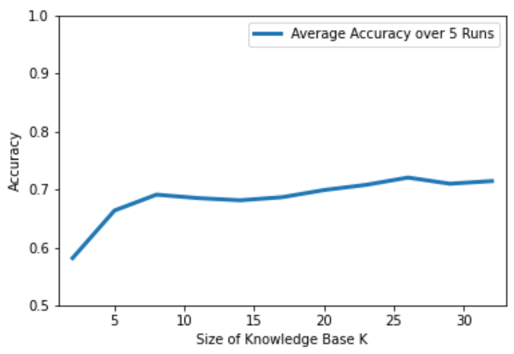


Figure 5.1: Performance over Different Size of Knowledge Base on **CUB**

5.2 Effect of Knowledge Base and Attention

Before each experiment, the size of the knowledge base, K, is pre-defined, such as K=10 in all of the experiments in this project. Even though the best knowledge base size is varying from different datasets, an empirical evaluation is run to show that the different knowledge base size does not lead to a significant difference in performance. We take the CUB dataset as an example. All the evaluation experiments are run under the same hyper-parameter setting, but they are different in the size of the knowledge base. We test ZSTL with a set of candidate sizes of the knowledge base. The candidate size starts from 2 to 33 with an interval of 3. Since the initialization of the knowledge base is randomly selected from the meta-training set and would lead to different performances, the results are the average accuracy of 5 different experiments. The effect of knowledge base size is shown in Figure 5.2. The performance is improved sharply at the beginning, then the performance would become stable after the knowledge base size around 10. Therefore, the knowledge base size is preset before the experiment, and we mainly focus on selecting the regularisation parameter during hyperparameter tuning.

Both Softmax and Sparsemax are used as the activation function in the attention mechanism. They both produce a discrete probability distribution over the latent task representations. Sparsemax produces more sparse solutions than Softmax. Therefore, Sparsemax proposes a stricter parameter sharing requirement among the learning tasks. Both Sparsemax and Softmax produce similar results in all experiments excepting the cold-start user recommendation problem in LastFM. The LastFM has the largest number of tasks among all the experiments in this project, thus it is more likely there is a pair of tasks that are completely different from each other. Softmax has already put the effort into preventing negative transfer by producing a low value in the probability distribution for those dissimilar latent task representations. However, each entry in softmax distribution is large than 0. There will still be mildly parameter sharing among dissimilar tasks. Sparsemax instead fixes this problem by producing 0 values for some the latent tasks and leads to better performance than Softmax in cold-start user recommendation in LastFM.

In ZSTL, we assume the tasks model for all the tasks in one experiment are identical. We train the matrix factorization model for the parameters in task models. Matrix factorization in the linear model has already been shown its success in zero-shot learning in [33, 34]. For DNN based non-linear model, we employ matrix factorization at every layer of the model. The bases for the parameters at every layer of the task model are the meta-knowledge (knowledge base) of the bi-level optimization based meta-learning framework. The knowledge base is updated based on the outer objectives with respect to the predicted task parameters. We also maintain a latent representation of task descriptors that allows novel tasks to uses its description to utilize the knowledge base to perform zero-shot learning. The latent task descriptor representation is updated by attention transfer [54] when the knowledge base is updated. Figure 5.2 visualizes the effectiveness of ZSTL in zero-shot learning for the DNN model. Figure 5.2 shows the model prediction of a sine wave regression task at the meta-testing set using predicted task parameters from inner objectives after 1, 500, 1500, and 2500 iterations of meta-knowledge update. We can observe that meta-knowledge can be used to perform zero-shot prediction with high accuracy after numerous iteration of meta-knowledge updates.

5.3 Conclusion and Future Work

In this project, we propose a method to perform zero-shot task transfer learning using bi-level optimization, meta-learning and attention mechanism. We deploy a matrix factorization on the parameters of the task model to learn a latent representation of task parameters. The latent representation of task parameter is regarded as meta-knowledge and updated by the outer objective of bi-level optimization during training. The attention mechanism provides an efficient

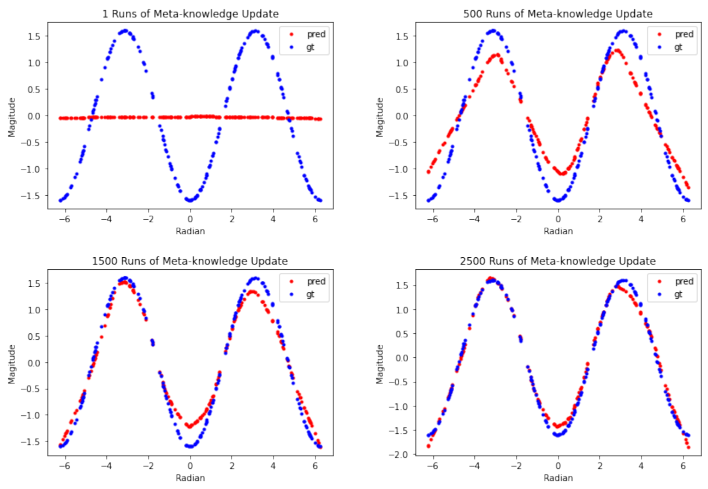


Figure 5.2: Meta-knowledge Update for Sine Wave Regression **Blue** is the ground truth label; **Red** is the predicted label

implementation to predict the parameter for different tasks. Another latent representation for task descriptor is trained to give an approximation to the dependency among task parameters and latent representation of task using attention transfer. The zero-shot transfer is achieved by combining the results from the attention mechanism and the latent representation of task parameters to estimate task parameters for unseen tasks. We test the proposed methods on two synthetic datasets and three real-world datasets. The proposed method can outperform the existing matrix factorization based method and achieve comparable results with the DNN based method.

The limitation of the proposed methods is that the task descriptions are not considered into predicting the task parameter for novel tasks but only used to approximate the dependency among tasks. Therefore, it is necessary to consider how to incorporate the task description to learn the task parameters for zero-shot task transfer learning. Besides, more complicated forms of task descriptions can be considered as well, including but not limited to texts, natural languages instructions as well as images. We could also consider zero-shot task transfer learning for more complicated task, such as complicated computer vision tasks beyond classification and time-series prediction models. Lastly, there could be a theoretical study on the effectiveness of matrix/tensor factorization in designing more efficient DNN model and improving the model's generalization and interpretability.

Bibliography

[1] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *NIPS*, 2006.

[2] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA, 2011. ACM.

[3] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem, 2018.

[4] Maurizio Corbetta and Gordon L Shulman. Control of goal-directed and stimulus-driven attention in the brain. In *Nature reviews. Neuroscience*, volume 3,3, pages 201–215, 2002.

[5] Manqing Dong, Feng Yuan, Lina Yao, Xiwei Xu, and Liming Zhu. Mamo: Memory-augmented meta-optimization for cold-start recommendation, 2020.

[6] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 109–117, New York, NY, USA, 2004. Association for Computing Machinery.

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017.

[8] Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. Forward and reverse gradient-based hyperparameter optimization. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of*

Machine Learning Research, pages 1165–1173, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[9] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1568–1577, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[13] Elyor Kodirov, Tao Xiang, and Shaogang Gong. Semantic autoencoder for zero-shot learning, 2017.

[14] Soheil Kolouri, Mohammad Rostami, Yuri Owechko, and Kyungnam Kim. Joint dictionaries for zero-shot learning, 2017.

[15] Kris Korrel, Dieuwke Hupkes, Verna Dankers, and Elia Bruni. Transcoding compositionally: using attention to find more generalizable solutions, 2019.

[16] Artus Krohn-Grimberghe, Lucas Drumond, Christoph Freudenthaler, and Lars Schmidt-Thieme. Multi-relational matrix factorization using bayesian personalized ranking for social network data. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, page 173–182, New York, NY, USA, 2012. Association for Computing Machinery.

[17] Abhishek Kumar and Hal Daumé. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 1723–1730, Madison, WI, USA, 2012. Omnipress.

[18] Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning, 2012.

[19] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. Melu: Meta-learned user preference estimator for cold-start recommendation, 2019.

[20] Jingjing Li, Mengmeng Jin, Ke Lu, Zhengming Ding, Lei Zhu, and Zi Huang. Leveraging the invariant side of generative zero-shot learning, 2019.

[21] Jingjing Li, Mengmeng Jing, Ke Lu, Lei Zhu, Yang Yang, and Zi Huang. From zero-shot learning to cold-start recommendation. In *AAAI*, 2019.

[22] Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1871–1880, 2019.

[23] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[24] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '18, page 1930–1939, New York, NY, USA, 2018. Association for Computing Machinery.

[25] André F. T. Martins and Ramón F. Astudillo. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1614–1623. JMLR.org, 2016.

[26] Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. Sparse coding for multitask and transfer learning. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, page II–343–II–351. JMLR.org, 2013.

[27] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning, 2016.

[28] Junhyuk Oh, Satinder Singh, Honglak Lee, and Pushmeet Kohli. Zero-shot task generalization with multi-task deep reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 2661–2670. JMLR.org, 2017.

[29] Arghya Pal and Vineeth N Balasubramanian. Zero-shot task transfer, 2019.

[30] Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. Zero-shot learning with semantic output codes. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1410–1418. Curran Associates, Inc., 2009.

[31] Parth Pathak, Mithun Das Gupta, Niranjan Nayak, and Harsh Kohli. Aqupr: Attention based query passage retrieval. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 1495–1498, New York, NY, USA, 2018. Association for Computing Machinery.

[32] Scott Reed, Zeynep Akata, Bernt Schiele, and Honglak Lee. Learning deep representations of fine-grained visual descriptions, 2016.

[33] Bernardino Romera-Paredes and Philip H. S. Torr. An embarrassingly simple approach to zero-shot learning. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 2152–2161. JMLR.org, 2015.

[34] Mohammad Rostami, David Isele, and Eric Eaton. Using task descriptions in lifelong machine learning for improved performance and zero-shot transfer. *J. Artif. Intell. Res.*, 67:673–704, 2020.

[35] Paul Ruvolo and Eric Eaton. Ella: An efficient lifelong learning algorithm. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, page I–507–I–515. JMLR.org, 2013.

[36] Suvash Sedhain, Aditya Menon, Scott Sanner, Lexing Xie, and Darius Braziunas. Low-rank linear cold-start recommendation from social data, 2017.

[37] Suvash Sedhain, Scott Sanner, Darius Braziunas, Lexing Xie, and Jordan Christensen. Social collaborative filtering for cold-start recommendations.

In *Proceedings of the 8th ACM Conference on Recommender Systems*, page 345–348, New York, NY, USA, 2014. Association for Computing Machinery.

[38] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017.

[39] Jivko Sinapov, Sanmit Narvekar, Matteo Leonetti, and Peter Stone. Learning inter-task transferability in the absence of target task samples. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '15, page 725–733, Richland, SC, 2015. International Foundation for Autonomous Agents and Multiagent Systems.

[40] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning, 2017.

[41] Richard Socher, Milind Ganjoo, Hamsa Sridhar, Osbert Bastani, Christopher D. Manning, and Andrew Y. Ng. Zero-shot learning through cross-modal transfer, 2013.

[42] Jae Woong Soh, Sunwoo Cho, and Nam Ik Cho. Meta-transfer learning for zero-shot super-resolution, 2020.

[43] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning, 2017.

[44] Manasi Vartak, Arvind Thiagarajan, Conrado Miranda, Jeshua Bratman, and Hugo Larochelle. A meta-learning perspective on cold-start recommendations for items. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6904–6914. Curran Associates, Inc., 2017.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[46] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning, 2016.

[47] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

[48] Yongqin Xian, H. Christoph Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning - a comprehensive evaluation of the good, the bad and the ugly. *TPAMI*, 2018.

[49] Yongqin Xian, Tobias Lorenz, Bernt Schiele, and Zeynep Akata. Feature generating networks for zero-shot learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5542–5551, 2018.

[50] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention, 2015.

[51] Yongxin Yang and Timothy M. Hospedales. A unified perspective on multi-domain and multi-task learning, 2014.

[52] Yongxin Yang and Timothy M. Hospedales. Deep multi-task representation learning: A tensor factorisation approach. *ArXiv*, abs/1605.06391, 2017.

[53] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. Abcnn: Attention-based convolutional neural network for modeling sentence pairs, 2015.

[54] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer, 2016.

[55] Amir R. Zamir, Alexander Sax, William Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[56] Jiejie Zhao, Bowen Du, Leilei Sun, Fuzhen Zhuang, Weifeng Lv, and Hui Xiong. Multiple relational attention network for multi-task learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '19, page 1123–1131, New York, NY, USA, 2019. Association for Computing Machinery.

[57] Zimu Zheng, Yuqi Wang, Quanyu Dai, Huadi Zheng, and Dan Wang. Metadata-driven task relation discovery for multi-task learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4426–4432. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[58] Ziwei Zhu, Shahin Sefati, Parsa Saadatpanah, and James Caverlee. Recommendation for new users and new items via randomized training and mixture-of-experts transformation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1121–1130, New York, NY, USA, 2020. Association for Computing Machinery.

Appendix A

Dependency of Project

A.1 Configuration

This section records the python version, the relevant third-party packages, and datasets necessary to run source code in [1].

- python 3.6.9
- pytorch 1.6.0+cu101
- Jupyter Notebook
- numpy 1.18.5
- pickle 4.0
- sklearn 0.22.2.post1
- tqdm
- matplotlib.pyplot
- sparsemax
  Using the implementation from [15]
  https://github.com/KrisKorrel/sparsemax-pytorch
- AwA2 http://cvml.ist.ac.at/AwA2/AwA2-features.zip

[1] https://github.com/Kaho-Lee/Zero-Shot-Task-Transfer-Learning

- CUB-200-2011 https://www.kaggle.com/veeralakrishna/200-bird-species-with-11788-images
- LastFM http://files.grouplens.org/datasets/hetrec2011/hetrec2011-lastfm-2k.zip

A.2 Comparison Methods Implementation

- TaDeLL
  Reimplementing TaDeLL in pytorch with enabled GPU based on the official implementation of the relevant work [35] from the same group.
  https://github.com/paulruvolo/ELLA
- UMTL
  Reimplementing UMTL from scratch based on the instruction on [51, 3]
- LoCo
  Reimplementing LoCo from scratch based on the instruction on [36]

A.3 How to run experiment?

1. Downloading the necessary dataset and/or generating synthetic data using the script " "Torch_SyntheticSine.ipynb" at Data_Proccessing folder at the source code repository
2. Using the scripts at Data_Proccessing folder to perform feature extraction and single task training to form the dataset for each tasks used at the experiment
3. The experiment results are from "ZSTL_xxx.ipynb" scripts at the source code repository. More details are available at the online repository.