Matching Networks with Semantic Attributes for One-Shot Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Despite the breakthroughs of deep neural networks (DNN), especially in computer vision, learning from a few examples (i.e. few-shot learning) still remains a challenge. Standard supervised learning models are data-demanding, which often require huge amount of iterative training. When new data is encountered, the model could hardly learn efficiently from the data. The combination of metric learning, deep neural features and augment neural networks with external memories fosters a novel architecture, Matching Network. However, the dataset $mini$ImageNet is far from enough to learn visual features effectively, resulting in unsatisfactory classification results on few-shot learning tasks. So we propose a method introducing semantic features as priori knowledge, to enhance the Matching Network classifier.

1 Introduction

Human are capable of learning new concepts after very few examples. Sometimes, we can even learn new concept through only one instance. For example, children can generalize the concept of "panda" from a single picture. Motivated by this, lots of researchers are devoted to the task: "one-shot" learning, which consists of learning a class from a single labelled example.

Deep learning has made tremendous progress in lots of fields such as reinforcement learning [1], localization [2] and classification [3]. However, these models often need a large dataset to learn new instances. Even with large dataset, DNN still needs lots of tricks like dropout, early stopping and data augmentation etc. to obtain a good result. So even with a powerful model like CNN, one-shot learning task is still a very hard task in machine learning. Existing one-shot learning methods can be divided to directed supervised learning-based methods and the transfer learning-based approaches. The directed ways, such as k-NN, aim to use the one-shot training set to do supervised training while the transfer learning follows the similar setting to zero-shot learning which is trying to transfer knowledge from auxiliary dataset to the target set. In this project, we have tried several ways from various perspective to attack the one-shot learning task.

Metric Learning: Sometimes a better representation of images will make the data easier to classify. There are many ways of learning useful representation like the Siamese Neural Networks [4] , which uses a symmetry net structure to find better representation of images. In our paper, we try to utilize a directed supervised method Large Margin Nearest Neighbor [4] to attack this task from the metric learning perspective.

Memory Mechanism: After using memory mechanism, the neural nets can function more like "computer". Some features of the examples can be stored in the memory. If both images share these features, the probability that they belong to the same class increases. These neural networks work

Submitted to 30th Conference on Neural Information Processing Systems (NIPS 2016). Do not distribute.

like computer, which can use heads to access memory. This approach was proposed by Deep Mind. Their model is called Memory-Augmented Neural network. In this project we also reimplement this MANN.[5]

Combination of Memory and Metric Learning: Deep Mind proposed a new network called Matching Network [6] (MN). They use metric learning to find a better representation of memory. And then use the memory mechanism to do one-shot classification task. Besides, they also pointed out a good training strategy which matches the training and test in the transfer learning setting. And this method produces very good results. Our work includes reimplementation of MN. Besides, we also tried to modify the structures of MN.

Semantic Attributes: Semantic attributes use priori knowledge in natural language processing or feature extractors to extract semantic attributes, which can be used to do one-shot learning and zero-shot learning task. [7] Besides the reimplementation work of papers, we proposed our own model called Matching Network with Semantic Attributes to help attention and memory mechanism to predict the label.

2 Related Work

2.1 Large margin nearest neighbour(LMNN)

LMNN uses semidefinite programming to learn a Mahanalobis distance in the kNN classification settings. The final goal for LMNN is that the k-nearest neighbors belong to the same class while keeping examples in different classes away with a large margin. The Mahanalobis distance is defined as

$$D_M(x_i, x_j) = (x_i - x_j)^T M (x_i - x_j)$$

where the matrix $M = L^T L$ is called as Mahanalobis Matrix . The Mahanalobis distance measures the dissimilarity of two vectors, x_i and x_j from the same distribution. The details of definition and optimization algorithms are discussed in [4] After the Mahanalobis matrix is found, it can be

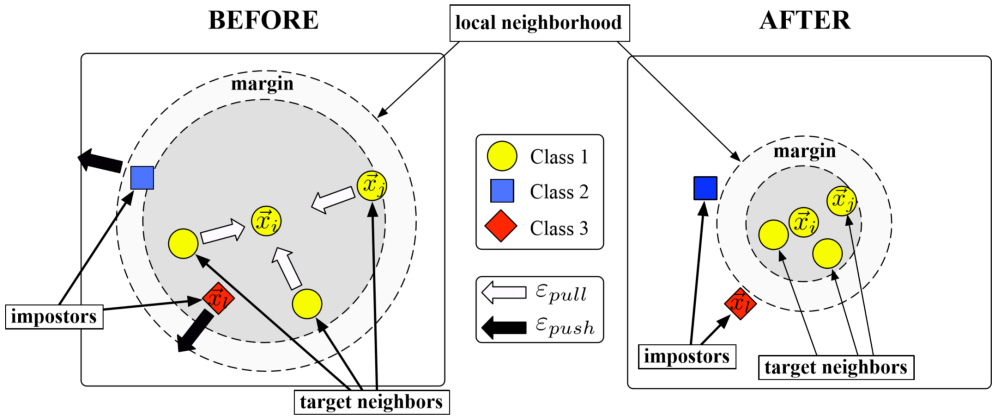

Figure 1: The examples belonging to the same class inside some margin will be pulled closer, while the ones in different classes would be pushed away.

projected from the original image space $X \in \mathbb{R}^{n \times d}$ to a new metric space XL^T. In this new feature space, the examples with the same label would be closer, while the ones with different labels are kept away. After utilizing the training data to learn a Mahanalobis matrix, we can project the test image X to new metric space and implement nearest neighbour algorithm in this new space.

2.2 NTM & MANN

Neural Turing Machine[8] consists of a controller such as a LSTM, which builds some connections between inputs and memory through read and write Heads. After taking in the inputs, the controller

exports keys, which are used to retrieve or stored as part of the memory contents, which is also called memory matrix. The memory content retrieved by the keys serves as the input to a classifier inside the controller. Read and write heads are implemented by introducing the concepts: location-based addressing and content-based addressing. The intuitive explanation is that location-based addressing can shift the heads left or right, pointing to the address of the memory contents, while in the content-based addressing the keys produced by the controller are used to calculate the similarities like cosine similarities with memory contents, which are used to retrieve contents according to its relative magnitude. Or it can be simply viewed as a kind of attention mechanism. Inspired by the Neural Turing Machine (NTM), DeepMind introduces NTM to one shot learning task which is called Memory-Augmented Neural Network (MANN)[5]. Unlike NTM using both location-based addressing and content-based addressing, MANN only uses content-based addressing.

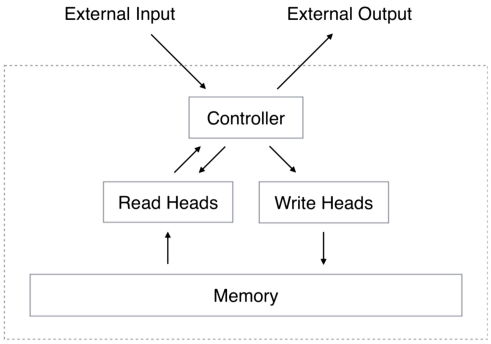

Figure 2: Neural Turing Machine Architecture. Controller takes in input images and outputs predictions during each training epoch. Controller also interacts with Memory using a set of heads.

3 Model

3.1 Matching Network (MN)

The first part of our model is mainly based on two methods mentioned in the paper"Matching Networks for One Shot Learning"[6]. First, given a support set $S = \{(x_i, y_i)\}_{i=1}^{k}$, our model defines a function c_S (or classifier) for each S, i.e. a mapping $S \rightarrow c_S(.)$, and a target set $T = \{(\hat{x}, \hat{y})\}$ to evaluate the performance of classifying c_S. Second, we employ a training strategy which is tailored for one-shot learning from the support set S.

3.1.1 Model Architecture

For the raw images, we trained a CNN model with just a few layers to extract rough features(g' for support set, $f'$for target set). Then we use the full context embedding method to modify the features f' and g' and then obtain f and g. Finally, we use the attention kernel to calculate the similarity(cosine or L2 norm) between target and support set, and obtain the prediction.

The Attention Kernel It is a kind of content-based attention mechanism based on similarity metrics, i.e. cosine similarities. If two features are similar according to the evaluation result, the two images are more likely to belong to the same class. To better represent features of images, the features extracted by NN are fed to attention mechanism which calculate the similarities between features of support sets and features of target sets. We obtain the attention by using softmax function on similarity:

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

After that, we can easily obtain the prediction by weighted summation:

$$\hat{y} = \sum_{i=1}^{k} a(\hat{x}, x_i) y_i$$

Fully Context Embeddings (FCE) The basic idea behind FCE is that when extracting features of an image, it is worth considering the features of the other images which may help form a better feature space. We use the output of the CNN model as input of a BiLSTM model to extract features of S, While for the features on T, we use a p-step LSTM with attention to extract features. We can extract the feature g from the BiLSTM easily, but for the p-step LSTM with attention (attLSTM)

$$f(\hat{x}, S) = attLSTM(f'(\hat{x}), g(S), p)$$

and each step can be implemented as follows (step i):

$$\begin{aligned} \hat{h}_i, c_i &= LSTM(f'(\hat{x}), [h_{i-1}, r_{i-1}], c_{i-1}) \\ h_i &= \hat{h}_i + f'(\hat{x}) \\ r_{i-1} &= \sum_{j=1}^{k} att(h_{i-1}, g(x_j)) g(x_j) \\ att(h_{i-1}, g(x_j)) &= softmax(h_{i-1}^T g(x_j)) \end{aligned}$$

note that it is a two-layer LSTM.

3.1.2 Training Strategy

In this model, we implemented a training strategy, that our model has to perform well with support sets S' which contain classes never seen during training. The criteria of training are derived from the idea of Meta-Learning, the goal of which is to learn a good learner for the specific situation. Considering the test procedure, the model should perform well with the support set S', in which the instances are from classes never seen during the training process.

So during the training process, "episodes" are created by the same way of instance sampling as is in the test procedure. Firstly, a label set L is sampled from the full train label set T. Then support set S is sampled with the labels in L, resulting n*k instances in the support set (for a n-way, k-shot task). A batch for 'testing' B is also generated. The learning task aims to minimize the error predicting the labels in the batch B, with the existence of support set S.

3.2 Semantic Feature Embedding

As we all know, the methods use semantic features to solve the image classification problems in zero shot learning tasks. Besides, the current matching network methods do not take the semantic features of images into consideration. From this perspective, we proposed a modified method based on MN. However, the $mini$Imagenet dataset do not have the semantic attribute tags as the AwA dataset. Under time and cost limitation, we can not label the $mini$Imagenet dataset manually. So we consider an automatic extraction method of semantic attributes.

In this project, we used an image visual-semantic embedding method proposed by Ryan Kiros et al.,[9]and they also proposed a modified method in 2017. They used different dataset to train the model including Flickr8K, Flickr30K and MS COCO. We used the model trained on the MS COCO dataset and extract the semantic vectors corresponding to images. MS COCO is a dataset with category labelling, instance spotting, instance segmantation and other natural language description of the images. In VSE model, we created an embedding space of multimodel(LSTM and CNN). With this embedding space, we can find the latent relationship between semantic vectors and image vectors. Finally we can extract an 1024 dimensions semantic feature vector for each new image in $mini$Imagenet.

3.3 MN with Semantic Attributes

For an image in the dataset, we can obtain the visual features (1600 dimentions) and further obtain semantic features (1024 dimentions) from our model. Then we consider a parameter to adjust the influence of both kinds of features:

$$pred = p * image\_feature + (1 - p) * semantic\_feature$$

. We can fine tune the parameter by validation dataset. In this way we can directly combine two kinds of features.

However, this method does not use the semantic features to benefit the training process of attention kernel mentioned in matching network. So we propose other methods to combine the features. After using the CNN layer to extract the image visual features, we combine the semantic features and the image visual features by connecting them together as the input of the FCE layer. Noted that the semantic features are fixed during the training process, so that we can fix the bias of visual feature extraction.

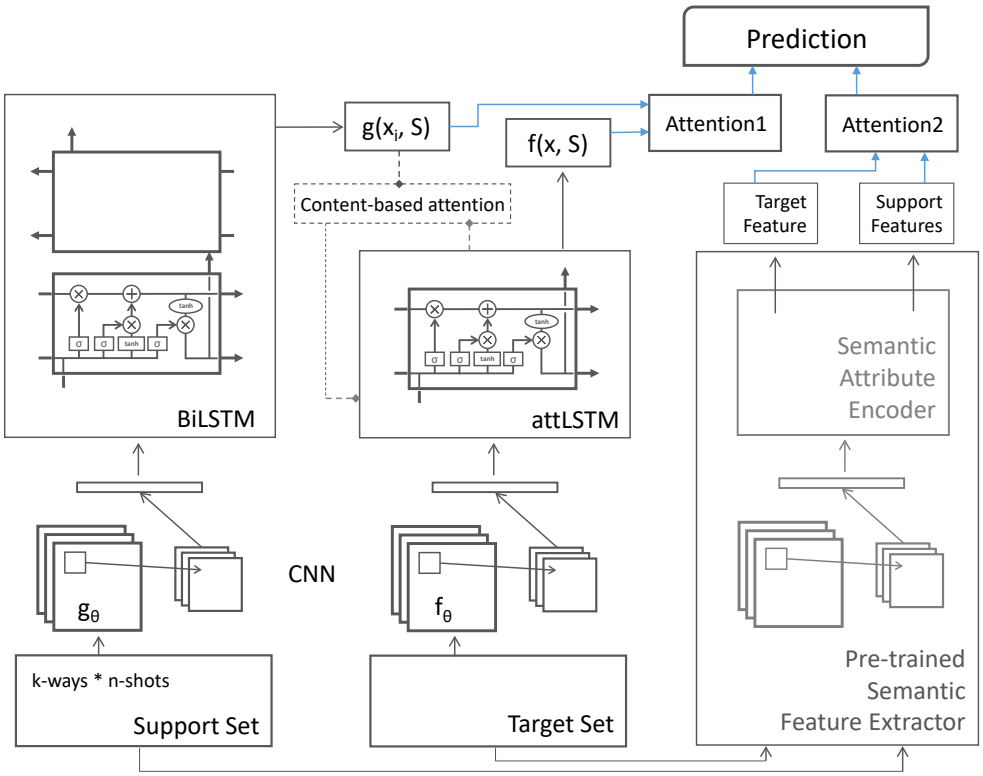

Figure 3: Architecture of Matching Network with Semantic Attributes. Note that the dashed lines stand for hidden states of attLSTM

4 Experiments

4.1 Evaluation Metrics

In our experiments on the models, we use *mini*ImageNet, created by DeepMind[6], as the dataset, which consists of 60,000 images in 100 different classes. The dataset is derived from the most famous computer vision dataset ImageNet. In this dataset, 80 classes are used for training and validation, and the remaining 20 classes for testing.

All of our experiments revolve around the same task: a N-way, k-shot learning task on *mini*ImageNet. Especially, one shot learning is on the condition of k=1. Models are provided with a set with k

labelled instances from each of N classes. Accordingly, the random performance is 1/N. As for direct supervised learning methods such as baseline models listed in table 1 and LMNN, we use only test set of $mini$ImageNet to implement the learning process. We first sample k labelled instances from each of N classes sampled from test classes for training, while keeping the rest for testing. For transfer learning methods MANN, MN and our model, we do training on train and validation sets, and test on the test sets.

4.2 Image Classification on $mini$Imagenet

For one shot learning, we implemented 6 baseline classifiers: k-NN, SVM-SVC, SVM-SVC with RBF kernel, decision tree and linear ridge classifier. The training strategy follows the above subsection. In one shot learning task, among all our baseline classifiers, k-NN outperforms all the other classifiers with the accuracy 29%. And our baseline model k-NN has outperforms the pixels model in [6]. As we can see, SVM results resemble the mathematical expectation of random performance. Bare neural network works even worse than random. We infer that the neural network has huge number of parameters. With only a small number of training data ,it is hard to learn. The LMNN can only reach an accuracy about 23%. In one shot learning, the k in LMNN must be set to 1 which means that it only pushes away one point that is near the center point, so the model cannot learn a better structure of the feature space. Hence, the accuracy is very low compared to k-NN. MANN can stand at 33.6% better than all of our baseline models.

For the Matching Network (MN) model, we implemented the original model mentioned in the paper, and we did get better performance than those models we mentioned above. However, we did not get the better performance than the paper mentioned. The reason may be a lack of fine tuning and parameter adjustment. The MN model with FCE reaches a better result, which confirms the attention mechanism. Noted that our final set of the parameters is as follows: batch size = 32, epoch=100, learning rate(initial)=10^{-3}, p=3(with FCE). The following models have the same initial settings.

As for our own models, we proposed two modified methods. First, considering the good performance of BiLSTM model in time series problems, we regard the p-step attLSTM as a time series, so we replace the attLSTM model with BiLSTM model. However, we do not get a good performance. As far as we are concerned, the reason may be the small value of p. BiLSTM model may not have outstanding performance on such task on the condition that the length of time series are small. Second, we use the semantic features to assist the prediction. As we can see, this model outperforms all the other models we have implemented. But we also discovered that the accuracy of the model are lower than the state of the art method. That may be caused by following reasons: the lack of fine tuning and parameter adjustment, and the relatively low accuracy of the VSE model, which limit the performance of our model.

Table 1: Results on $mini$ImageNet

	Model	**Matching Fn**	**5-way Accuracy** 1-shot	5-shot
Baseline Models	NEURAL NETWORK CLASSIFIER	Cosine	19.9%	20.4%
	SVM-SVC	L2Norm	20.0%	20.0%
	SVM-SVC-RBFKERNEL	L2Norm	20.0%	20.0%
	DECISION TREE CLASSIFIER	Cosine	23.5%	27.7%
	LINEAR RIDGE CLASSIFIER	Cosine	28.0%	32.4%
	K-NEAREST NEIGHBOURS	L2Norm	29.0%	33.0%
Implemented Models	LMNN	Cosine	23.0%	—
	MANN	Cosine	33.6%	—
	MATCHING NETWORKS	Cosine	40.1%	48.6%
	MATCHING NETWORKS(FCE)	Cosine	41.3%	49.1%
Our Models	MATCHINGNET+BARE BILSTM	Cosine	35.2%	41.3%
	MATCHINGNET+SEMANTIC FEATURE	Cosine	42.7%	52.3%

4.3 Analysis and Visualization

4.3.1 Training Process

In our implementation, we calculated the test accuracy on test dataset, when the x‘model performs better than before on validation dataset. According to the figure, we can conclude that the test accuracy may not improve any more, because the test loss increases while the training and validation loss decrease. So we can early stop this training process, which is a normal strategy.

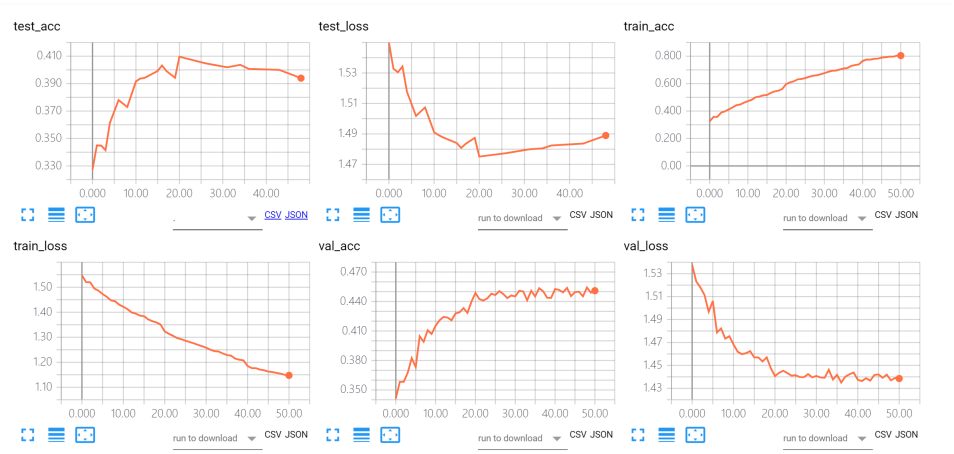

Figure 4: The visualization of training process on the MN model .

4.3.2 Visualization via t-SNE

t-Distributed Stochastic Neighbour Embedding (t-SNE) is a widely used technique for dimensionality reduction, to visualize high-dimensional datasets. We used t-SNE to visualize the features extracted from the Matching Network, including both with Full Context Embedding(FCE) and without FCE. Besides the comparison of the features above, we also implemented t-SNE on the features of different stages on the VSE model, namely the 1600-dimension visual features, and further 1024-dimension semantic features. The t-SNE results of both stages are illustrated in figure 4. As we can see in the

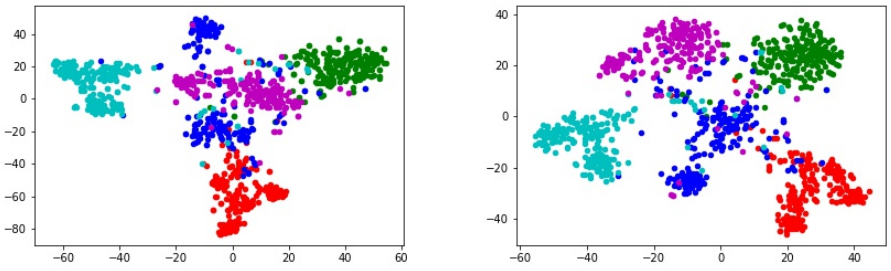

Figure 5: t-SNE illustration on features extracted from VSE, with visual features on the left, and semantic features on the right.

left figure, the output feature of well-trained CNN shows a good separation of the five instances. However, the blue points in the figure are separated and overlapping with other dots. In the right figure, the blue points are almost joint together. The good nature of the extractor motivated us to use VSE to enhance the original Matching Network. The model is trained on Training set with validation, and with test accuracy of 48.6 and 49.1 on 5-way 5-shot task. In Figure 5, the subfigures both show the ill nature of separation of the feature extractor in the Matching Networks. However, in the figure on the right, the magenta dots gathers well, while some of the red points gathers well too, this may explain the difference between the results of Matching Network with and without FCE.

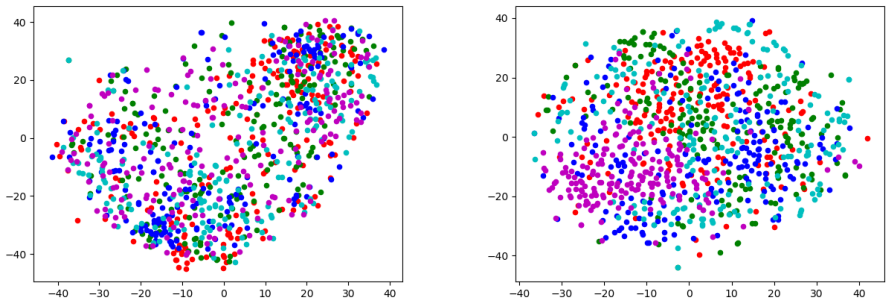

Figure 6: t-SNE illustration on features extracted from Matching Networks, the feature on the right is extracted through FCE , while the left is not.

4.4 Environment

GPU: Nvidia GeForce GTX 1080

Software Environment: Ubuntu 17.01, Python 3.6

5 Conclusion

In this project, we have reimplemented several state of the art works and proposed several modified methods based on previous work. The matching network with semantic features we have proposed is limited by the accuracy of the semantic attributes encoder. To improve the method, we can use the knowledge graph to embed the entities, attributes and the relations between them, so that we can discover the latent relationship between visual features, semantic features and knowledge graph embedding features by multimodal space similar to the VSE model[9]. In this way, we may not only solve the image classification problem but also can solve image inference and image description problems.

References

References

[1] J. X. Chen, "The evolution of computing: Alphago," *Computing in Science and Engineering*, vol. 18, no. 4, pp. 4–7, 2016.

[2] W. Huang, H. Ding, and G. Chen, "A novel deep multi-channel residual networks-based metric learning method for moving human localization in video surveillance," *Signal Processing*, vol. 142, pp. 104–113, 2018.

[3] M. Duan, K. Li, C. Yang, and K. Li, "A hybrid deep learning CNN-ELM for age and gender classification," *Neurocomputing*, vol. 275, pp. 448–461, 2018.

[4] K. Q. Weinberger, J. Blitzer, and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," in *Advances in Neural Information Processing Systems 18 [Neural Information Processing Systems, NIPS 2005, December 5-8, 2005, Vancouver, British Columbia, Canada]*, pp. 1473–1480, 2005.

[5] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. P. Lillicrap, "One-shot learning with memory-augmented neural networks," *CoRR*, vol. abs/1605.06065, 2016.

[6] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain* (D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, eds.), pp. 3630–3638, 2016.

[7] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 453–465, 2014.

[8] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," *CoRR*, vol. abs/1410.5401, 2014.

[9] R. Kiros, R. Salakhutdinov, and R. S. Zemel, "Unifying visual-semantic embeddings with multimodal neural language models," *CoRR*, vol. abs/1411.2539, 2014.