Deep Learning at Alibaba

Rong Jin
Alibaba Group, Hang Zhou, China
jinrong.jr@alibaba-inc.com

Abstract

In this talk, I will focus on the applications and the latest development of deep learning technologies at Alibaba. More specifically, I will discuss (a) how to handle high dimensional data in deep learning and its application to recommender system, (b) the development of deep learning models for transfer learning and its application to image classification, (c) the development of combinatorial optimization techniques for DNN model compression and its application to large-scale image classification and object detection, and (d) the exploration of deep learning technique for combinatorial optimization and its application to the packing problem in shipping industry. I will conclude my talk with a discussion of new directions for deep learning that are under development at Alibaba.

1 Introduction

The last decade has witnessed great success of deep learning in multiple domains, including speech recognition, image classification, and video content analysis. Despite the amazing progress, we also run into many challenges when coming to practical applications of deep learning technologies. In this talk, we would like to share some of the key developments of deep learning at Alibaba that explicitly address limitations of the existing deep learning techniques.

The first challenge is how to deal with very high dimensional but sparse data in deep learning. This is because typically, in deep learning, we need to learn either a conventional layer or a fully connected layer that maps the input data into a lower dimensional representation. When coming to the representation of hundreds of millions of features, it becomes computationally expensive to learn such a mapping layer, making it difficult to fully explore the power of deep learning. In Section 2, we address this challenge by developing a novel framework for deep learning that explicitly handles the computational challenge with very high dimensional data.

The second challenge is related to transfer learning. Although numerous studies are devoted to exploring deep learning technique for transfer learning, they usually assume a relatively small difference between the source domain and the target domain. In contrast, for many applications of transfer learning at Alibaba, a very significant difference exists between the source domain and the target domain. As a result, a simple fine tune based approach did not work well. To this end, in Section 3, we propose to learn an explicit transform, from limited data, that explicitly relates instances from the source domain to those from the target domain.

The third challenge is how to compress a large complicated deep learning model into a smaller one without losing its prediction power. This has been a popular topic in the recent studies of deep learning. To reduce the complexity of deep learning model, a common approach is to search for a neural network with discrete weights. Although multiple algorithms have been developed to compress large deep learning models, they often failed to deliver the desire performance when coming to construct neural networks with weights represented by only two or three bits. In Section 4, we address this challenge by developing an optimization technique, based on the idea of extra gradient descent, that can significantly facilitate the search of a neural network with discrete weights.

The last subject to be addressed in this talk is how to explore reinforcement learning techniques for discrete optimization. Although the idea of exploring neural network techniques in combinatorial optimization is not new at all, it has not generated exciting results, mostly due to the limited performance of neural network. We explore machine learning techniques for combinatorial optimization by learning an appropriate search policy from the long term awards. In Section 5, we will share some of the promising results when applied the reinforcement learning technique to 3D bin packing problems.

In Section 6, we will conclude this talk by looking at the latest development of deep learning at Alibaba.

2 Deep Learning with Very High Dimensional Inputs

Alibaba Group has one of the world's largest online shopping platforms. The recommender system plays an important role at Alibaba e-commerce platform because it is able to display the most valuable items that fit in the needs of individual customers. To make an accurate estimation of users' needs at any moment, an effective online recommender system needs to take into account user profiles, the context information of the scenarios, and the real-time feedback collected from indi-

vidual users. In addition, most recommender systems have to ensure the diversity of displayed items in order to maximize user experience.

Generalized linear models (e.g. logistic regression) are widely used in online recommender systems, where consumers are often represented by their ages, genders, living locations and user group ids, while merchants are usually described by their categories, sellers, brands and item ids. Since most features are represented by one-hot coding, we often end up with a representation of hundreds of millions of binary features in order to accurately predict the probability for an user to click on a specific item in a given scenario.

Despite the great success of generalized linear model for modeling click through rate (CTR), it often failed to yield accurate prediction for items with limited sales [Cheng *et al.*, 2016]. This is because feature engineering is usually well designed for those popular items but not for items with limited sales. For instance, a cross-product transformation is often introduced to model the interaction between the binary features of user and merchants. But, due to the limitation of memory, only the cross-products with respect to popular items are computed for the feature vector, while cross-product features are left out for those unpopular items. We address this challenge by proposing a deep learning framework that is able to handle the very high dimensional input vectors. Instead of manually introducing the cross-products among input features, we apply deep learning to automatically model the nonlinear dependence among different input features.

The main challenge for the proposed deep learning framework is how to deal with the high dimensional sparse vectors since most deep learning approaches are not designed to handle billions of input features. We address this challenge by introducing a random coding scheme that maps the high dimensional input vector into one with relatively low dimension. The main advantages of using a random coding scheme are (a) it dramatically reduces the dimensionality of the input vectors, (b) it is computationally efficient, and (c) with a high probability, it preserves the geometric relationship among high dimensional vectors. Using the encoded dense vectors, we apply a multi-layer non-linear transformation to generate appropriate vector representation for users and items. These learned representations will finally be fed into a linear prediction model to estimate the click through rate. Figure 1 illustrates the overall idea of the proposed deep learning framework.

We run the online experiments to verify the effectiveness of the proposed deep learning framework. The test scenario is to estimate both CTR and CVR for the displayed items returned by our search engine, which are used to rerank the returned items. We compare the ranking results for the proposed deep learning framework to those generated by the linear model (i.e. logistic regression model). The A/B tests show that, using the proposed method, we observe a 6% improvement in both CVR and GMV compared to directly using the linear model. The improvement is more significant during the single's day in 2016 (i.e. 11/11/2016, the single largest promotion day at Alibaba): we observed more than 16% improvement in GMV, and close to 20% improvement in CTR. We note that the key difference between the proposed framework and a generalized linear model is that a linear model is unable to account for the nonlinear interaction between any user and any item. We believe that it is this difference that limits the performance of generalized linear model.

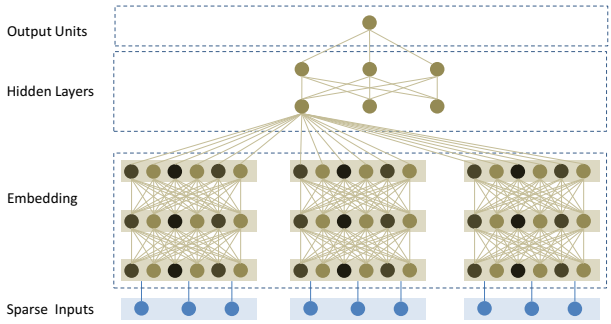

Figure 1: The architecture of the proposed deep neural network for very high dimensional sparse inputs

3 Deep Transfer Learning

Despite significant progress made in transfer learning, most existing approaches of transfer learning explored the fact that only a small difference exists between the source domain and the target domain. For the real-world challenges we deal with at Alibaba, we often observe a significant gap between source domain and target domain, making it difficult for a simple fine tune approach to deliver the desirable performance. To address this challenge, we propose to learn an explicit transform from limited data, that directly maps deep feature learned from the source domain to compact features in the target domain. Using the learned transform, we are able to successfully apply transfer learning techniques to tasks such as image recognition, retrieval, and matching effectively.

More specifically, we first learn a deep convolutional network from the data of the source domain that outputs the feature mapping. We then learn a linear transform layer $f(x) = Wx$, where W is the transformation matrix, that maps the embedding features x of the source domain into the target domain. The optimal transform is obtained by minimizing the distance between images with similar tags and at the same time maximizing the distance between images with different tags. The overall framework is given in Figure 2. A CNN model is learned from the source domain to output a vector representation for each image. A transformation layer is then learned to transfer the feature vectors output from the pre-trained CNN network into a vector representation for the target domain.

A triplet loss based deep learning is developed to learn the optimal transform that effectively combines the strength of metric learning with the power of CNN [Schroff *et al.*, 2015]. Let x_i^a and x_i^p be the feature vectors output from the pre-trained neural network for the query image and the image "similar" to the query [1], respectively. Let x_i^n be the feature vector for the image that share a different tag from that of the

[1] An image is similar to a given query if both of them are labeled by the same tag

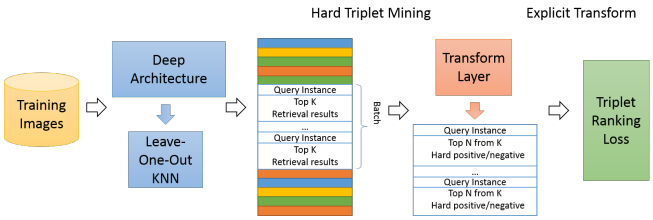

Figure 2: The transfer learning framework that combines the strength of metric learning, based on triplet loss, with the power of CNN. Hard triplet mining is used to identify the most informative subset of triplets in order to improve the learning efficiency.

query. Using vectors x_i^a, x_i^p, and x_i^n, we form a triplet and define the triplet loss as follows

$$\left[\|f(x_i^a) - f(x_i^p)\|_2^2 + 1 - \|f(x_i^a) - f(x_i^n)\|_2^2\right]_+$$

where $[z]_+$ is the hinge loss that outputs one when $z > 0$ and zero otherwise. By minimizing the triplet loss, we require that the distance between x_i^a and x_i^p is significantly smaller than that between x_i^a and x_i^n. As a result, our overall objective is to minimize the sum of the triplet loss over all the possible triples that can be formed, i.e.

$$\sum_{i=1}^{N} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 + 1 - \|f(x_i^a) - f(x_i^n)\|_2^2\right]_+$$

One challenge with optimizing the sum of triplet losses is that it is infeasible to compute the loss for all triplets given its size is $O(n^3)$. To address this challenge, we only select the difficult triplets, i.e. *hard triplets*, where x_i^n is visually similar to the given query x_i^a but with different tags. To further improve the learning efficiency, we iteratively update the set of *hard triplets* in each epoch, which is more effective than using a set of static triplets. More specifically, for each query image, we keep its K nearest neighbors, with $K = 200$. The hard triplets are identified at each epoch by selecting the negative instances x_i^n from the KNN of x_i^a that have smaller distance than those of the positive instances.

Our empirical studies have shown that this approach is significantly more effective than randomly sample a subset of triplets to form the the objective function. The pre-train model is learned from the product images collected by the Taobao platform. The target domain in our experiment is images from OpenImages dataset [Krasin *et al.*, 2016]. We run our algorithm over the Aliyun ODPS platform with 2000 cores to train the model, which took 2 hours for each epoch. By using the transfer learning with hard triplet mining strategy, we are able to improve the classification accuracy by 6.9% when compared to the original pre-trained features. In contrast, we only observe 2% improvement for the classification accuracy when using simple random selection strategy to form the triplets.

4 Learning a Deep Network with Discrete Weights

The success of deep learning largely owes to the fast development of computing resources. Most of the deep learning models are deployed on high-ended GPUs or CPU clusters. On the other hand, deeper networks typically impose heavy storage footprint due to the enormous amount of network parameters. The complexity of deep neural network, both in terms of computation and storage requirements, has made it difficult to run deep learning algorithms for scenarios with limited memory and computational resources.

In this talk, we present a unified framework for low-bits quantized neural networks that leverage the alternating direction method of multipliers (ADMM) [Boyd *et al.*, 2011], which was originally designed for convex optimization. We first model the quantized neural network as a optimization problem with discrete constraints. By introducing consensus constraints, we find such problem may be solved by the ADMM algorithm, and we furthermore introduce special optimization approaches to address the subproblems related to the ADMM algorithm.

Denote $f(W)$ as the loss function of a normal neural network, where $W = \{W_1, W_2, \cdots, W_L\}$ and W_i is the parameter of the i-th layer in the network. Low-bits quantized neural network can be formulated as a constrained optimization problem:

$$\begin{aligned} \min_{W} \quad & f(W) \\ \text{s.t.} \quad & W \in \mathcal{C} \end{aligned} \tag{1}$$

where $\mathcal{C}$ is a discrete set including numbers that are either zero or powers of two. The advantage of restricting the weights to zero or powers of two is that an expensive floating-point multiplication operation can then be replaced by a sequence of cheaper and faster binary bit shift operations. Since batch normalization [Ioffe and Szegedy, 2015] is often used to improve the convergence of deep neural network training, it will lead to a scale invariance in the weight parameters. As a result, we further introduce a scaling factor α to the constraints, i.e., instead of requiring $\mathcal{C} = \{\cdots, -2, -1, 0, +1, +2. \cdots\}$, we simply restrict $\mathcal{C}$ to $\mathcal{C} = \{\cdots, -2\alpha, -\alpha, 0, +\alpha, +2\alpha, \cdots\}$ with an arbitrary scaling factor $\alpha > 0$ that is strictly positive [Rastegari *et al.*, 2016]. The optimization problem in (1) can then be rewritten as

$$\begin{aligned} \min_{W,G} \quad & f(W) + I_{\mathcal{C}}(G) \\ \text{s.t.} \quad & W = G \end{aligned} \tag{2}$$

where we introduce G to handle the discrete weights and the consensus constraint $W = G$ to ensure the resulting weight W will be discrete. Notation $I_{\mathcal{C}}$ is introduced to represent the indicator function, i.e.

$$I_{\mathcal{C}}(W) = \begin{cases} 0 & \text{if } W \in \mathcal{C}\,, \\ \infty & \text{if } W \notin \mathcal{C}. \end{cases}$$

Following the framework of ADMM, we need to optimize the augmented Lagrange of (2), i.e.

$$L_\rho(W, G, \lambda) = f(W) + I_{\mathcal{C}}(G) + (\rho/2)\|W - G\|^2 + \langle W - G, \lambda \rangle \tag{3}$$

where λ denotes the Lagrangian multipliers. Following the standard process of ADMM, this problem can be solved by repeating the following iterations:

$$W^{k+1} := \arg\min_{W} L_\rho(W, G^k, \lambda^k) \quad (4)$$

$$G^{k+1} := \arg\min_{G} L_\rho(W^{k+1}, G, \lambda^k) \quad (5)$$

$$\lambda^{k+1} := \lambda^k + \rho(W^{k+1} - G^{k+1}) \quad (6)$$

In order to efficiently run the ADMM algorithm, we have to solve the optimization problems in (4) and (5) efficiently. To this end, we develop an extra gradient descent method to effectively solve the optimization problem in (4) and an iterative quantization method to effectively solve the optimization problem in (5).

Accuracy	Binary	Ternary	{-4, +4}	Full Precision
Top-1	0.611	0.635	0.665	0.665
Top-5	0.838	0.852	0.875	0.871

Table 1: Accuracy of Resnet-18 on ImageNet classification

mAP	{-4, +4}	Full Precision
Darknet+SSD	0.624	0.642
VGG16+SSD	0.776	0.778

Table 2: mAP of VGG and Darknet on Pascal VOC 2007

We run the proposed algorithm against the imagenet dataset to discretize the weights of Resnet-18 [He *et al.*, 2016] into one, two and three bits, which corresponds to the titles of "Binary", "Ternary", and $\{-4, +4\}$ in Table 1. We measure the prediction accuracy of the top 1 and top 5 returned results in Table 1, and compare them to the performance of the original Resnet-18 with continuous weights. It is clear that when the number of bits increases to three, we observe almost no lost in prediction accuracy. We run the proposed algorithm for object detection, using either Darknet+SSD [Liu *et al.*, 2016] or VGG16+SSD, over Pascal VOC 2007. Table 2 summarize the results of the proposed approach and the original network. We again observe that with three bits of weight quantization, the resulting networks yield performances that are close to the original ones with continuous.

5 Learning to Optimize

Combinatorial optimization has found applications in many fields such as artificial intelligence, machine learning, operations research, mathematics, auction theory, and software engineering. Since many combinatorial optimization problems are NP-hard (e.g. traveling salesman problem, bin packing, and vehicle routing problem), solving them in practice often relies on handcrafted heuristics that help find approximate but competitive solutions efficiently. To address the limitation of using human-engineered heuristics, one approach is to design hyper-heuristics that leverages machine learning techniques in order to guide the search in the space of heuristics [Burke *et al.*, 2013]. The alternative approach is to operate in the solution space directly. Recent advances in sequence-to-sequence model [Sutskever *et al.*, 2014] have motivated the study of exploring deep neural network for combinatorial optimization [Vinyals *et al.*, 2015; Bello *et al.*, 2017].

In this work, we followed this line of work and applied deep learning and reinforcement learning methods to solve the 3D bin packing problem, an NP hard but also an important combinatorial optimization problem that has found applications in computational resource allocation and logistics (e.g. see [Coffman *et al.*, 1980; Chen *et al.*, 1995; Crainic *et al.*, 2008; Clautiaux *et al.*, 2014]). We developed a heuristic (a constructive approach) to obtain a competitive solution efficiently in practice. There are three key decisions which are heuristically made during the packing procedure: 1) the packing order of items; 2) the location where to place items; 3) the orientation of items to be placed. We show that appropriate heuristics can be learned by a pointer network and reinforcement learning method. Below, we will first introduce the pointer network framework and reinforcement learning for optimization, and then describe the preliminary results.

Pointer network Similar to [Bello *et al.*, 2017], the used pointer network consists of two recurrent neural network (RNN) modules (encoder and decoder) as shown in Figure 3. The input of this network is a sequence of dimensions of items to be packed, while the output is the sequence of packing (e.g. order of packing items).

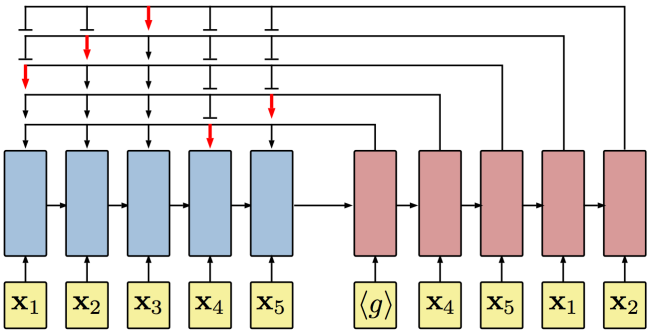

Figure 3: Pointer network(left: encoder, right: decoder) [Bello et al., 2017]

Reinforcement learning Given a sequence of packed items, we can obtain the smallest bin that can pack all the items. Thus, the surface area can be used to evaluate the sequence (the output of network). Denote the input sequence as s, and the output sequence as o, then the surface area is defined as $L(o|s) = WL + LH + WH$, where W, H, L are the dimensions of the smallest bin. A model-free policy based reinforcement learning model was used to optimize the parameters of the pointer network θ. Let $p(o|s)$ be the probability of choosing the packing sequence o given the input s. The training objective for a given s is the expected surface area, which is given as below

$$J(\theta|s) = \mathcal{E}_{o\sim p_\theta(\cdot|s)} L(o|s). \quad (7)$$

The total training objective is defined as $J(\theta) = \mathcal{E}_{s\sim\mathcal{S}} J(\theta|s)$, where s is sampled from a distribution $\mathcal{S}$. We used a training procedure similar to [Bello *et al.*, 2017] to optimize the parameters of the network.

Preliminary results We have carried out empirical studies to evaluate the performance of this learning to optimization model. For simplicity, we only used the learning to optimization framework to optimize the packing order, while the location and orientation of packed item are still decided by the existing heuristics. $150,000$ train samples and $150,000$ test samples are used for all experiments, where the number of packed items varies from 8, 10, to 12. The model was trained using the Adam optimizer. When the number of items to be packed is small (i.e. 8), we measure the optimality gap for the heuristic solutions, which is about 10% on average. For all experiments (item numbers are 8, 10, and 12), we found that the learning to optimize model achieves about 5% improvement over the heuristics in terms of average surface area [Hu *et al.*, 2017], a significant improvement given that all the heuristics used in the search are well tuned for the 3D bin pack problems.

6 Deep Learning at Alibaba: Latest Development

The recent studies of deep learning tend to examine neural networks with extremely large number of layers. For instance, in the recent reports of imagenet competition, the winning team has developed a neural network with over $1,000$ layers. Despite the encouraging performance, the extremely deep network has clear disadvantages: they are usually difficult to train and expensive to deploy. As an alternative solution to the extremely deep neural network, we ask if it is possible to introduce the high nonlinearity into the prediction function through the construction of complex activation functions, instead of the depth of neural network. In other words, we aim to develop a relatively shallow network but with very complex activation functions. This idea is inspired by the theory of function approximation that was first developed by Allen Pinkus [Pinkus, 1999]. According to Theorem 7.1 in [Pinkus, 1999], there exists an activation function σ which is C^∞, strictly increasing, and sigmoidal, and has the following property: for any continuous function $f \in C[0,1]^n$, and any accuracy bound $\varepsilon > 0$, there exist constants d_i, $c_{i,j}$, $\theta_{i,j}$, and γ_i, and vectors $w^{i,j} \in \mathbb{R}^n$, such that

$$\left| f(x) - \sum_{i=1}^{4n+3} d_i \sigma \left(\sum_{j=1}^{2n+1} c_{i,j} \sigma(\langle w^{i,j}, x\rangle + \theta_{i,j}) + \gamma_i \right) \right| \leq \varepsilon$$

In other words, this theorem implies that there exists a complex activation function of sigmoid type such that any continuous function can be well approximated by a neural network with two hidden layers. The advantage of searching for a nonlinear activation function is that activation function is an univariate function whose optimization can be done effectively even in the non-parametric setting [Tsybakov, 2008]. Encouraged by this theoretic result, within Alibaba, we are working toward the direction of optimizing the activation function within a two hidden layer network.

References

[Bello *et al.*, 2017] Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. In *ICLR*, 2017.

[Boyd *et al.*, 2011] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[Burke *et al.*, 2013] Edmund K Burke, Michel Gendreau, Matthew Hyde, Graham Kendall, Gabriela Ochoa, Ender Özcan, and Rong Qu. Hyper-heuristics: A survey of the state of the art. *Journal of the Operational Research Society*, 64(12):1695–1724, 2013.

[Chen *et al.*, 1995] CS Chen, Shen-Ming Lee, and QS Shen. An analytical model for the container loading problem. *European Journal of Operational Research*, 80(1):68–76, 1995.

[Cheng *et al.*, 2016] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pages 7–10. ACM, 2016.

[Clautiaux *et al.*, 2014] François Clautiaux, Mauro DellAmico, Manuel Iori, and Ali Khanafer. Lower and upper bounds for the bin packing problem with fragile objects. *Discrete Applied Mathematics*, 163:73–86, 2014.

[Coffman *et al.*, 1980] Edward G Coffman, Jr, Michael R Garey, David S Johnson, and Robert Endre Tarjan. Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM Journal on Computing*, 9(4):808–826, 1980.

[Crainic *et al.*, 2008] Teodor Gabriel Crainic, Guido Perboli, and Roberto Tadei. Extreme point-based heuristics for three-dimensional bin packing. *Informs Journal on computing*, 20(3):368–384, 2008.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[Hu *et al.*, 2017] Haoyuan Hu, Xiaodong Zhang, Xiaowei Yan, Longfei Wang, and Yinghui Xu. Solving a new 3d bin packing problem with deep reinforcement learning method. *Technical report*, 2017.

[Ioffe and Szegedy, 2015] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[Krasin *et al.*, 2016] Ivan Krasin, Tom Duerig, Neil Alldrin, Andreas Veit, Sami Abu-El-Haija, Serge Belongie, David Cai, Zheyun Feng, Vittorio Ferrari, Victor Gomes, Abhinav Gupta, Dhyanesh Narayanan, Chen Sun, Gal

Chechik, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*, 2016.

[Liu *et al.*, 2016] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*, 2016.

[Pinkus, 1999] Allan Pinkus. Approximation theory of the mlp model in neural networks. *Acta Numerica*, 8:143–195, 1999.

[Rastegari *et al.*, 2016] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, 2016.

[Schroff *et al.*, 2015] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 815–823, 2015.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[Tsybakov, 2008] Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Publishing Company, Incorporated, 1st edition, 2008.

[Vinyals *et al.*, 2015] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *NIPS*, pages 2692–2700, 2015.