DEEP METRIC LEARNING USING TRIPLET NETWORK

Elad Hoffer
Department of Electrical Engineering
Technion Israel Institute of Technology
ehoffer@tx.technion.ac.il

Nir Ailon
Department of Computer Science
Technion Israel Institute of Technology
nailon@cs.technion.ac.il

ABSTRACT

Deep learning has proven itself as a successful set of models for learning useful semantic representations of data. These, however, are mostly implicitly learned as part of a classification task. In this paper we propose the *Triplet network* model, which aims to learn useful representations by distance comparisons. We show promising results demonstrating the success of this model on various datasets. We also discuss future possible usages as a framework for unsupervised learning.

1 INTRODUCTION

For the past few years, deep learning models have been used extensively to solve various machine learning tasks. One of the underlying assumptions is that deep, hierarchical models such as convolutional networks create useful representation of data (Bengio (2009); Hinton (2007)), which can then be used to distinguish between available classes. This quality is in contrast with traditional approaches requiring engineered features extracted from data and then used in separate learning schemes. Features extracted by deep networks were also shown to provide useful representation (Zeiler & Fergus (2013a); Sermanet et al. (2013)) which can be, in turn, successfully used for other tasks (Razavian et al. (2014)).

Despite their importance, these representations and their corresponding induced metrics are often treated as side effects of the classification task, rather than being explicitly sought. There are also many interesting open question regarding the intermediate representations and their role in disentangling and explaining the data (Bengio (2013)). Notable exceptions where explicit metric learning is preformed are the *Siamese Network* variants (Bromley et al. (1993); Chopra et al. (2005); Hadsell et al. (2006)), in which a contrastive loss over the metric induced by the representation is used to train the network to distinguish between similar and dissimilar *pairs* of examples. A contrastive loss favors a small distance between pairs of examples labeled as similar, and large distances for pairs labeles dissimilar. However, the representations learned by these models provide sub-par results when used as features for classification, compared with other deep learning models. Siamese networks are also sensitive to calibration in the sense that the notion of similarity vs dissimilarity requires a notion of context. For example, a person might be similar to another person when a dataset of random objects is provided, but might need to be deemed as a dissimilar object when we wish to distinguish between two individuals in a set of individuals only. In this work, such a calibration is not required.

We follow a similar task to that of Chechik et al. (2010). For a set of samples $\mathbb{P}$ and a chosen rough similarity measure $r(x, x')$ given through a training oracle (e.g how close are two images of objects semantically) we wish to learn a similarity function $S(x, x')$ induced by a normed metric. Unlike Chechik et al. (2010)'s work, our labels are of the form $r(x, x_1) > r(x, x_2)$ for triplets x, x_1, x_2 of objects. Accordingly, we try to fit a metric embedding and a corresponding similarity function

satisfying:

$$S(x, x_1) > S(x, x_2), \ \ \forall x, x_1, x_2 \in \mathbb{P} \ \ \text{for which } r(x, x_1) > r(x, x_2).$$

In our experiment, we try to find a metric embedding of a multi-class labeled dataset. We will always take x_1 to be of the same class as x and x_2 of a different class, although in general more complicated choices could be made. Accordingly, we will use the notation x^+ and x^- instead of x_1, x_2. We focus on finding an L_2 embedding, by learning a function $F(x)$ for which $S(x, x') = \|F(x) - F(x')\|_2$. Inspired from the recent success of deep learning, we will use a deep network as our embedding function $F(x)$.

2 TRIPLET NETWORK

A *Triplet network* (inspired by "Siamese network") is comprised of 3 instances of the same feed-forward network (with shared parameters). When fed with 3 samples, the network outputs 2 intermediate values - the L_2 distances between the embedded representation of two of its inputs from the representation of the third.
If we will denote the 3 inputs as x, x^+ and x^-, and the embedded representation of the network as $Net(x)$, the final output will be the vector:

$$TripletNet(x, x^-, x^+) = \begin{bmatrix} \|Net(x) - Net(x^-)\|_2 \\ \|Net(x) - Net(x^+)\|_2 \end{bmatrix} \in \mathbb{R}^2_+$$

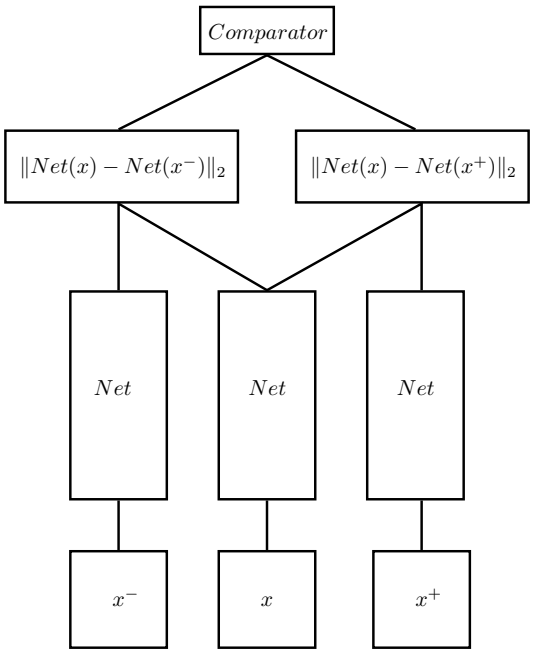

Figure 1: Triplet network structure

2.1 TRAINING

Training is preformed by feeding the network with samples where, as explained above, x and x^+ are of the same class, and x^- is of different class. The network architecture allows the task to be expressed as a 2-class classification problem, where the objective is to correctly classify which of x^+ and x^- is of the same class as x. We stress that in a more general setting, where the objective might be to learn a metric embedding, the label determines which example is *closer* to x. Here we simply interpret "closeness" as "sharing the same label". In order to output a comparison operator from the model, a SoftMax function is applied on both outputs - effectively creating a ratio measure. Similarly

to traditional convolutional-networks, training is done by simple SGD on a negative-log-likelihood loss with regard to the 2-class problem. By using the same shared parameters network, we allow the back-propagation algorithm to update the model with regard to all three samples simultaneously.

3 TESTS AND RESULTS

The Triplet network was implemented and trained using the Torch7 environment (Collobert et al. (2011)).

3.1 DATASET

The Triplet Network was trained on the *Cifar10* dataset (Krizhevsky & Hinton (2009)) which consists of 60000 32x32 color images of 10 classes (of which 50000 are used for training only, and 10000 for test only). No data augmentation or whitening was applied, and the only preprocessing was a global normalization to zero mean and unit variance.
Each training instance is a uniformly sampled set of 3 images, 2 of which are of the same class (x and x^+), and the third (x^-) is of a different class. Each training epoch consists of 640,000 such instances (randomly chosen each epoch), and a fixed set of 64,000 instances used for test. We emphasize that each test instance involves 3 images from the set of 10000 images which was excluded from training.

3.2 EMBEDDING NET

The embedding net used is a convolutional network, consisting of 3 convolutional and 2x2 max-pooling layers, followed by a fourth convolutional layer. A *ReLU* non-linearity is applied between two consecutive layers. Network configuration (ordered from input to output) consists of filter sizes $\{5,3,3,2\}$, and feature map dimensions $\{3,64,128,256,128\}$ where a 128 vector is the final embedded representation of the network. Usually in convolutional networks, a subsequent fully-connected layer is used for classification. In our net this layer is removed, as we are interested in a feature embedding only.

3.3 RESULTS

Training is done by SGD, with initial learning-rate of 0.1 that is manually reduced when test accuracy saturates. We used a momentum value of 0.9. We also used the Dropout regularization technique to avoid over-fitting. After training for 30 epochs, the network reached a fixed error of 6% (error is over triplet comparisons). We then used the embedding network to extract features from the full dataset. By using a simple multi-class SVM model with these representations, an accuracy of 84% on the test set was achieved for the full 10-class classification task. This result is comparable to state-of-the-art results with deep learning models, without using any artificial data augmentation (Zeiler & Fergus (2013b); Goodfellow et al. (2013); Lin et al. (2013)). We conjecture that data augmentation techniques (such as translations, mirroring and noising) may provide similar benefits to those described in previous works.
Another side-affect noticed, is that the representation seems to be sparse - about 25% non-zero values. This is very helpful when used later as features for classification both computationally and with respect to accuracy, as each class is characterised by only a few non zero elements.

3.4 2D VISUALIZATION OF FEATURES

In order to examine our main premise, which is that the network embeds the images into a representation with meaningful properties, we use PCA to project the embedding into 2d euclidean space which can be easily visualized (figure 2). We can see a significant clustering by semantic meaning, confirming that the network is useful in embedding images into the euclidean space according to their content.
Similarity between objects can be easily found by measuring the distance between their embedding and, as shown in the results, can reach high classification accuracy using a simple subsequent linear classifier. For convenience, we plot the confusion matrix (figure 3) for the classification results on the test set, which should be compared with its visual embedding (figure 2).

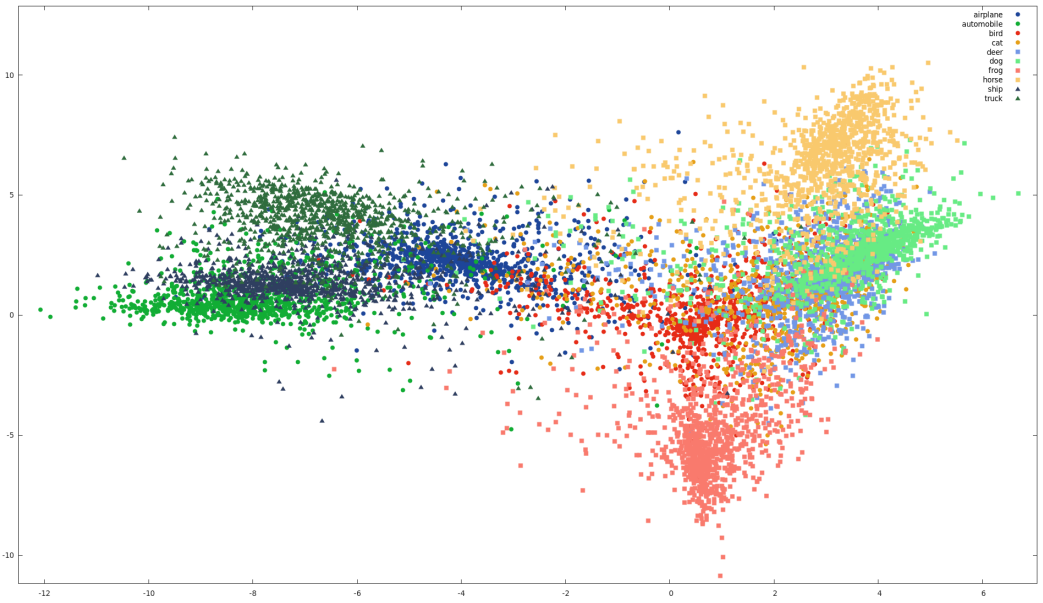

Figure 2: Euclidean 2d representation of embedded test data

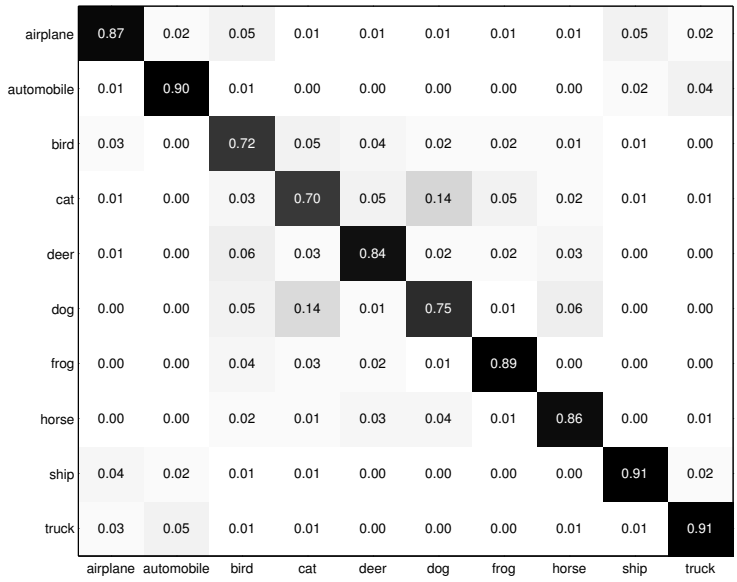

Figure 3: Confusion matrix on Cifar10 test set (numbers indicate accuracy)

4 FUTURE WORK

As the Triplet net model allows learning by comparisons of samples instead of direct data labels, usage as an unsupervised learning model is possible. Future investigations can be performed in several scenarios:

- **Using spatial information.** Objects and image patches that are spatially near are also expected to be similar from a semantic perspective. Therefore, we could use geometric distance between patches of the same image as a rough similarity oracle $r(x, x')$, in an unsupervised setting.
- **Using temporal information.** The same is applicable to time domain, where two consecutive video frames are expected to describe the same object, while a frame taken 10 minutes later is less likely to do so. Our Triplet net may provide a better embedding and improve on past attempts in solving classificaion tasks in an unsupervised encironment, such as that of (Mobahi et al. (2009)).

It is also well known that humans tend to be better at accurately providing comparative labels. Our framework can be used in a crowd sourcing learning environment. This can be compared with Tamuz et al. (2011), who used a different approach. Furthermore, it may be easier to collect data trainable on a Triplet network, as comparisons over similarity measures are much easier to attain (pictures taken at the same location, shared annotations, etc).

5 CONCLUSIONS

In this work we introduced the *Triplet network* model, a tool that uses a deep network to learn useful representation explicitly. The results shown on Cifar10 provide evidence that the representations that were learned are useful to classification in a way that is comparable with a network that was trained explicitly to classify samples. We believe that enhancement to the embedding network such as Network-in-Network model (Lin et al. (2013)), Inception models (Szegedy et al. (2014)) and others can benefit the Triplet net similarly to the way they benefited other classification tasks. Considering the fact that this method requires to know only that two out of three images are sampled from the same class, rather than knowing what that class is, we think this should be inquired further, and may provide us insights to the way deep networks learn in general. We have also shown how this model learns using only comparative measures instead of labels, which we can use in the future to leverage new data sources for which clear out labels are not known or do not make sense (e.g hierarchical labels).

ACKNOWLEDGMENTS

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan-Z GPU used for this research.

REFERENCES

Bengio, Yoshua. Learning Deep Architectures for AI, 2009. ISSN 1935-8237.

Bengio, Yoshua. Deep learning of representations: Looking forward. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7978 LNAI, pp. 1–37, 2013. ISBN 9783642395925. doi: 10.1007/978-3-642-39593-2_1.

Bromley, Jane, Bentz, James W, Bottou, Léon, Guyon, Isabelle, LeCun, Yann, Moore, Cliff, Säckinger, Eduard, and Shah, Roopak. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688, 1993.

Chechik, Gal, Sharma, Varun, Shalit, Uri, and Bengio, Samy. Large scale online learning of image similarity through ranking. *The Journal of Machine Learning Research*, 11:1109–1135, 2010.

Chopra, Sumit, Hadsell, Raia, and LeCun, Yann. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pp. 539–546, 2005. ISBN 0769523722. doi: 10.1109/CVPR.2005.202.

Collobert, Ronan, Kavukcuoglu, Koray, and Farabet, Clément. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, number EPFL-CONF-192376, 2011.

Goodfellow, Ian J, Warde-Farley, David, Mirza, Mehdi, Courville, Aaron, and Bengio, Yoshua. Maxout networks. *arXiv preprint arXiv:1302.4389*, 2013.

Hadsell, Raia, Chopra, Sumit, and LeCun, Yann. Dimensionality reduction by learning an invariant mapping. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, volume 2, pp. 1735–1742. IEEE, 2006.

Hinton, Geoffrey E. Learning multiple layers of representation, 2007. ISSN 13646613.

Krizhevsky, Alex and Hinton, Geoffrey. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep*, 2009.

Lin, Min, Chen, Qiang, and Yan, Shuicheng. Network in network. *CoRR*, abs/1312.4400, 2013. URL http://arxiv.org/abs/1312.4400.

Mobahi, Hossein, Collobert, Ronan, and Weston, Jason. Deep learning from temporal coherence in video. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 737–744. ACM, 2009.

Razavian, Ali Sharif, Azizpour, Hossein, Sullivan, Josephine, and Carlsson, Stefan. CNN Features off-the-shelf: an Astounding Baseline for Recognition. *Arxiv*, 2014. URL http://arxiv.org/abs/1403.6382.

Sermanet, Pierre, Eigen, David, Zhang, Xiang, Mathieu, Michael, Fergus, Rob, and LeCun, Yann. OverFeat : Integrated Recognition , Localization and Detection using Convolutional Networks. *arXiv preprint arXiv:1312.6229*, pp. 1–15, 2013. URL http://arxiv.org/abs/1312.6229.

Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. URL http://arxiv.org/abs/1409.4842.

Tamuz, Omer, Liu, Ce, Belongie, Serge, Shamir, Ohad, and Kalai, Adam. Adaptively learning the crowd kernel. In Getoor, Lise and Scheffer, Tobias (eds.), *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pp. 673–680, New York, NY, USA, June 2011. ACM. ISBN 978-1-4503-0619-5.

Zeiler, Matthew D and Fergus, Rob. Visualizing and Understanding Convolutional Networks. *arXiv preprint arXiv:1311.2901*, 2013a. URL http://arxiv.org/abs/1311.2901.

Zeiler, Matthew D and Fergus, Rob. Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*, 2013b.