Knowledge Transfer in Deep Convolutional Neural Nets

Steven Gutstein, Olac Fuentes and **Eric Freudenthal**
Computer Science Department
University of Texas at El Paso
El Paso, Texas, 79968, U.S.A.

Abstract

Knowledge transfer is widely held to be a primary mechanism that enables humans to quickly learn new complex concepts when given only small training sets. In this paper, we apply knowledge transfer to deep convolutional neural nets, which we argue are particularly well suited for knowledge transfer. Our initial results demonstrate that components of a trained deep convolutional neural net can constructively transfer information to another such net. Furthermore, this transfer is completed in such a way that one can envision creating a net that could learn new concepts throughout its lifetime.

Introduction

For any inductive learner, it is necessary to select an appropriate bias. This allows sufficient generalization of a target concept based upon a reasonably sized set of training examples. An insufficiently biased learner will have an overly large hypothesis space to search and will be prone to overtraining. An overly biased learner will be able to learn only a poor approximation of the true target concept. A good bias will enable a learner to successfully acquire a concept with fewer training examples and greater generality.

One bias that humans seem to have is that similar tasks employ similar solutions. It is generally accepted that people transfer knowledge acquired from previously learned tasks to master new ones. This transfer enables us to acquire new concepts quickly and accurately based on very few examples, because we have already learned to distinguish between relevant and irrelevant features.

There are several machine learning techniques that attempt to transfer relevant knowledge. They include discriminability-based transfer (Pratt 1993), multitask learning (Caruana 1997), explanation based neural nets (Thrun 1995), knowledge based cascade correlation (Schultz & Rivest 2000) and internal representation learning (Baxter 2000).

In this paper, we examine knowledge transfer for deep convolutional neural nets by using internal representation learning. We show that when a generalized internal representation has been achieved, new concepts can be learned with smaller training sets and nets with smaller capacities. However, as the size of training set increases, net capacity becomes more important relative to transferred knowledge.

Background & Related Work

For almost all image recognition problems, shift-invariance and moderate insensitivity to both rotations and geometric distortions are important biases. There are three basic approaches to creating these invariances in a neural net. These approaches are:

1. Exhaustive training - give examples of all permutations of the desired invariant parameter(s)
2. Pre-processing - pre-process all input to remove the invariant parameter(s)
3. Choosing a specific class of network architecture - create a network that is insensitive to the invariant parameter(s)

Network architectures that restrict neurons to interact only with local receptive fields from the layer beneath them, and that require corresponding weights from all the receptive fields of a given layer to be equal have been successfully used for over 15 years to achieve these invariances (Bishop 1995).

This method is employed by Deep Convolutional Neural Nets (DCNN's) (LeCun *et al.* 1999). Although these nets are usually referred to as just Convolutional Neural Nets, we refer to them as Deep Convolutional Neural Nets, in order to equally emphasize the use of a deep architecture as well as locally receptive fields. Unlike the creators of this architecture, our primary interest is knowledge transfer.

DCNN's organize the neurons of a given layer into several feature maps. The neurons composing a given feature map will all share weights and common receptive fields. Each such map may, therefore, be viewed as acting to detect a given feature, wherever it occurs. Furthermore, at higher layers of the net, feature maps may be regarded as identifying the combinations of various low-level features that compose more complex higher-level features. The presence of feature maps as an architectural component of DCNN's makes these nets an attractive candidate for knowledge transfer, since these maps represent discrete, localizable detectors for specific features that distinguish among the various classes.

There are two standard types of layers found in a DCNN: convolutional and sub-sampling layers. In the convolutional layers (C-layers) of the net, each feature map is constructed by calculating the convolution of one or more small learned kernel over the feature maps of the previous layer, or over the original input, when it constitutes the previous layer. Although a single feature map may be connected to many feature maps of the prior level, it is connected to each by an individual learned kernel. This results in a feature map that will reflect the presence of a particular local feature, or local combination of features, wherever they occur in maps in the prior layer. It is this convolution of small kernels that gives DCNNs an architecturally based bias for translational invariance, which is useful for problems with strong local correlations.

In the sub-sampling layers (S-layers), each feature map is connected to exactly one feature map of the prior layer. A kernel of the sub-sampling layers is not convolved over the corresponding feature map of the prior layer. Instead, the input feature map is divided into contiguous non-overlapping tiles, which are the size of the kernel. Each sub-sampling kernel contains two learnable parameters:

1. a multiplicative parameter, which multiples the sum of the units in a given tile, and
2. an additive parameter, which is used as a bias.

This gives DCNNs a decreased sensitivity to minor rotations and distortions of an image, which helps make them robust with respect to unimportant variations.

Multi-Task Learning (MTL) involves simultaneously training a net in several related tasks. It has been shown to improve performance by allowing several tasks to share information during training (Caruana 1997). Because the architecture of DCNN's forces the early layers to act as feature extractors, these nets should make good use of MTL, because different tasks may require recognition of the same features located in different parts of an image for different classes. Furthermore, as the upper layers of the net are required to encode more instances from a set of classes (e.g. specific characters from a set of all characters), they should be learning internal representations of those classes, which could provide an effective mechanism for dimensionality reduction.

With this in mind, it is convenient to view a DCNN as possessing two halves - a lower half, which acts as a feature extractor and an upper half, which combines the features to produce a reduced dimension representation of the input image. Once a DCNN has been trained to recognize a number of specific classes from a set of related classes (i.e. characters, faces etc.) it should be possible to train to recognize other related classes by only training weights in the upper layers of the net. These upper layers will, in essence, be solving a problem with significantly reduced dimensionality. This will now give three main advantages:

1. Significantly faster training time, because the dimensionality of the problem domain has decreased
2. Better generalized accuracy with small training sets, since some learning obtained from previous training sets is retained.
3. Smaller expected difference between the expected errors of the training set and testing set, due to the decreased net capacity.

Using weights from a trained neural net to seed weights in an untrained net bears a marked similarity to Discriminability-Based Transfer, first introduced by Pratt (Pratt 1993). Pratt took a classical feed-forward neural net (i.e. 1 input layer, 1 hidden layer and 1 output layer) and trained it to perform one task. Then, he used some of the learned weights to seed a new net with the same architecture, which was trained in a new, related task. However, direct transfer of all the original weights (i.e. literal transfer) was found to be counter-productive.

In order to identify which neurons aided learning and which ones hindered learning, Pratt looked at the response of each hidden neuron to the training set for the new class. Those neurons that provided significant information gain (as measured by mutual information) about the class of the various training inputs had their input weights transferred. Those that did not had their input weights reset. All weights between neurons of the hidden and output layers were reset. This differs from our approach in that:

1. Claims were made only with respect to the speed of training, not the accuracy
2. We used the topology of a multi-layered net to help determine which weights should be transferred. So, there is no need for Pratt's pre-training processing
3. The nets used are classical feed-forward nets. Because DCNN's have feature maps distributed over many layers, we can infer that feature maps closer to the input layer will identify simple, low level features, while those deeper into the net will detect progressively higher level features. So, the lower the level of a feature map, the more likely it is to be transferable

The method of Explanation Based Neural Nets (EBNN's) is very different than ours (Thrun 1995). It trains a single net using training samples of the form $(x, f(x), \nabla f(x))$. The pair $(x, f(x))$ corresponds to a standard supervised learning training example. However, $\nabla f(x)$ is produced by a second neural net, which has been trained to calculate a 'comparator' function.

A comparator function merely reports whether or not two inputs are of the same class, but not to which class either belongs, whereas the EBNN is being trained to recognize when a given input is a member of a particular class. For example, given two faces, the comparator net will recognize whether or not the two faces are the same. However, the EBNN is being trained to recognize whether or not the given face belongs to a specific individual (e.g. Fred). Because the comparator net provides information gleaned from other related tasks, knowledge transfer is occurring. In order to then incorporate this information, EBNN's are trained using TangentProp (Simard *et al.* 2001).

A recent use of a pure comparator function for recognition or verification of a large number of categories was demonstrated by Chopra et al. (Chopra, Hadsell, & LeCun 2005). In this work, a siamese net was trained on a relatively small

number of faces to recognize whether a given pair of faces were from the same person. This technique was then able to correctly label pairs of faces, which came from people not seen during training, as being same or different. The main difference between this technique and ours is that whereas ours concentrates on learning a robust internal representation, Chopra et. al. concentrate upon learning a similarity metric. It is our hope that by concentrating on creating robust internal representations using progressively higher order features, our technique will enable nets to transfer relevant knowledge across a wider range of tasks.

Knowledge Based Cascade Correlation(KBCC) (Schultz & Rivest 2000) is an extension of Fahlman & Lebiere's (Fahlman & Lebiere 1990) Cascade Correlation. Of all the methods mentioned here, this is the only one that allows the topology of a neural net to change as learning occurs. The modification employed by KBCC is to allow whole trained networks to be absorbed into the developing net during training. Although our method does not do this, it seems likely that feature maps generated in one learning task could be profitably transferred to others. So, a DCNN could present a rich source of sub-nets for KBCC.

The approach of using a neural net to learn an internal representation of each related class was employed by Bartlett & Baxter (Baxter 2000). This is the most similar to our approach, since it depends upon the neural net to determine which features should be extracted, reduces those features to a relatively small dimensional space (as compared to the original input) and then uses the resultant encoding to provide the solution to various boolean tasks (i.e. is the input an example of class A, class B, class C etc.?).

The primary difference in our approach is that by using a DCNN not only are some important biases built into the net's architecture, but also because many more distinct feature maps are used, it is possible to envision efficiently combining our technique with KBCC to transfer knowledge even further afield (e.g. character recognition would seem to involve the ability to recognize outlines, line intersections, angles etc. It is quite plausible that more general recognition problems also make use of these abilities). By learning internal representations, Baxter & Bartlett (Bartlett & Baxter 1998) report being able to reliably categorize characters their net hadn't been explicitly trained to recognize. An error rate of 7.5% was achieved on the 2618 characters that their net had not been specifically trained to identify.

We have not yet duplicated this accomplishment. We believe this is primarily due to the fact that we used a smaller number of classes to provide us with knowledge transfer. Bartlett & Baxter (Bartlett & Baxter 1998) point out that to truly learn an internal representation, one needs to be exposed to many classes. For this reason, they chose to use Japanese Kanji as an Optical Character Recognition (OCR) test bed of his technique. Their net learned an internal representation using 400 classes of characters. We used English characters and learned an internal representation with only 20 classes of characters. Additionally, Baxter and Bartlett exposed their net to the new characters during training, by classifying them all as being 'other' (i.e. not one of the 400 training characters), we have not exposed our net in any way to the new characters, prior to specifically attempting to learn to recognize them. Nevertheless, we demonstrated significant improvements in learning by using an internal representation, particularly with very small training sets for new classes.

Knowledge Transfer in DCNN's

Our approach takes advantage of several architectural features of DCNN's. First, the feature maps of the various levels represent neurons that can be functionally grouped together. This is because they may be considered as roughly locating the same feature(s) in a given image. Second, the layered structure would suggest that feature maps at the lower levels of the net respond to simple features, whereas maps at higher levels respond to higher-level features. By the time the upper-most levels are reached, the feature maps consist of a single unit indicating a particular combination of features. These may reasonably be relied upon to provide a reduced dimension representation of the original image.

This implies that when a DCNN has been trained to identify several classes from a set of classes (i.e. several characters from the set of all characters), a new DCNN can be trained to identify another subset of that same (or possibly just a similar) set of classes, using the already trained lower layers of the first net to initialize itself. If those lower layers are treated as fixed, the smaller capacity of the net being trained will result a smaller expected difference between the testing an training errors. Furthermore, by fixing the weights of the transferred neurons, one can begin to think about having a net that was trained to recognize members of one subset of classes, being further trained to recognize an additional subset of classes. In effect, this net could continue to learn new concepts, either by learning new combinations of existing feature maps or by the judicious introduction of new feature maps.

Our experiments represent an initial effort to determine the extent to which we can transfer knowledge between DCNN's in this fashion.

Experiments

Our experiments used the LeNet5 style architecture (LeCun *et al.* 1999), which has already been successfully used for OCR. The main difference between our net, shown in Figure 1, and LeNet5 is that our last layer is a combination of LeNet5's F6 and Output layer. It consists of 20 units instead of 10 and is referred to as F6.

The dataset used was the NIST Special Database 19, which contains 62 classes of handwritten characters corresponding to '0'-'9', 'A'-'Z' and 'a'-'z'.

The net was initially trained to recognize a set of 20 characters. The entire net was trained on a set of 20 characters using 400 samples of each. Each character was assigned a 20 bit random vector. The random bit assignment provides a greater distance between target vectors than a standard '1 of N' encoding. The resultant net was trained and achieved an accuracy of 94.38% on this training set. This net will be referred to as the 'source' net, since it will be the source of

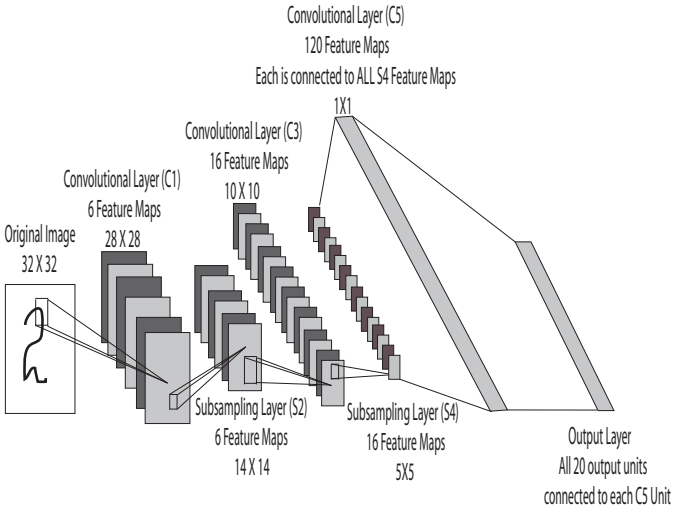

Figure 1: Architecture of our net, which is a slightly modified version of LeNet5 (LeCun *et al.* 1999). It should be noted that the 'feature maps' in layers C5 & F6 are 1 neuron x 1 neuron, which means they could with equal accuracy be considered as traditional neurons in a non-weight sharing feed-forward neural net

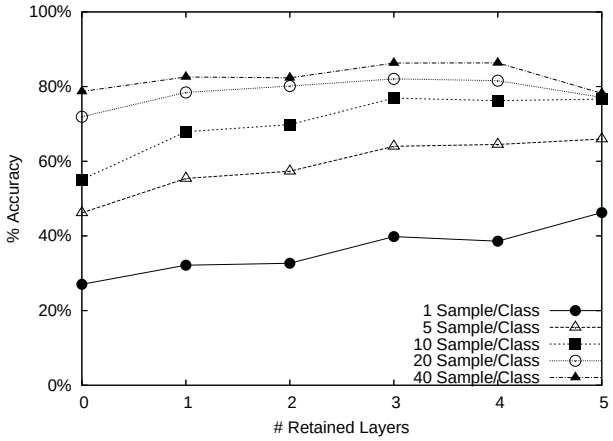

Figure 2: Comparison of learning curves showing accuracy vs. number of retained levels for various numbers of samples per class in the training set. Curves show, from top to bottom, results for 40, 20, 10, 5 and 1 sample per class. Each point represents the average of 5 trials on a testing set with 1,000 character samples.

our transferred knowledge. Frequently, much larger training sets are used to obtain near perfect accuracy (Simard, Steinkraus, & Platt). However, for our purposes, this accuracy was deemed sufficient and perhaps even more appropriate than near perfect, since we wouldn't want to 'over-transfer' knowledge.

Next, we attempted to use some of the acquired knowledge to aid in learning to recognize a new set of 20 characters. These new characters were also assigned 20 bit random target vectors. Then, the weights from the bottom n layers of the source net were copied over to the new net, where $0 \leq n \leq 5$. Transferred weights were kept fixed and not allowed to change during training. To find the best choice for n, we ran a series of experiments beginning with $n = 5$ and culminating with $n = 0$. This last scenario, of course, corresponds to the absence of any knowledge transfer. Were we to have tried allowing $n = 6$, that would correspond to transferring all the weights from the source net and not allowing any training. For obvious reasons, we did not do this.

The performance of each net was evaluated using a testing set comprised of 1,000 characters. We ran 5 learning trials for each value of n. Additionally, we experimented with training sets of 1, 5, 10, 20 and 40 samples/class. Results are shown in figures 2-5.

As each layer is released to be retrained, more free parameters become available, thus increasing the capacity of the net. However, the increase in free parameters is very sharply spiked at the 5th layer of the net. In fact, this is where more than 90% of the net's free parameters lie. The number of free parameters for each layer is shown in table 1.

So, when only the top level has not been retained, the net can only train with 4.6% of the free parameters normally

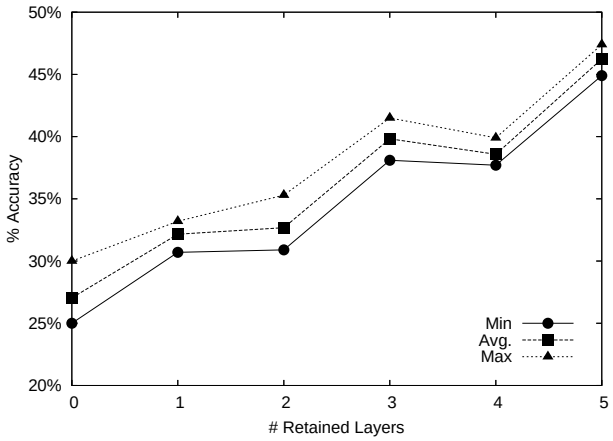

Figure 3: Comparison of learning curves showing accuracy vs. number of retained levels for 1 sample per class in the training set. Curves show minimum accuracy, average accuracy and maximum accuracy obtained over 5 trials on a testing set with 1,000 character samples

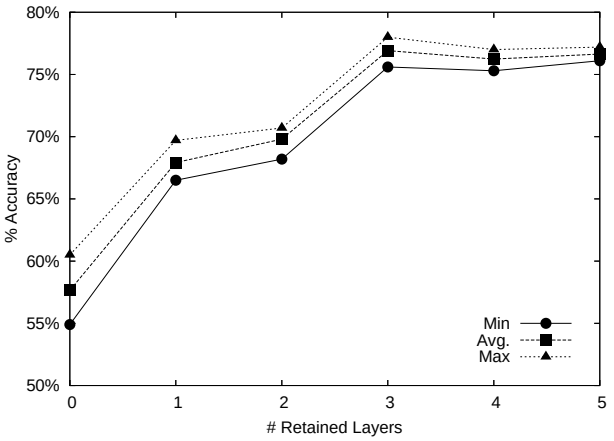

Figure 4: Comparison of learning curves showing accuracy vs. number of retained levels for 10 samples per class in the training set. Curves show minimum accuracy, average accuracy and maximum accuracy obtained over 5 trials on a testing set with 1,000 character samples

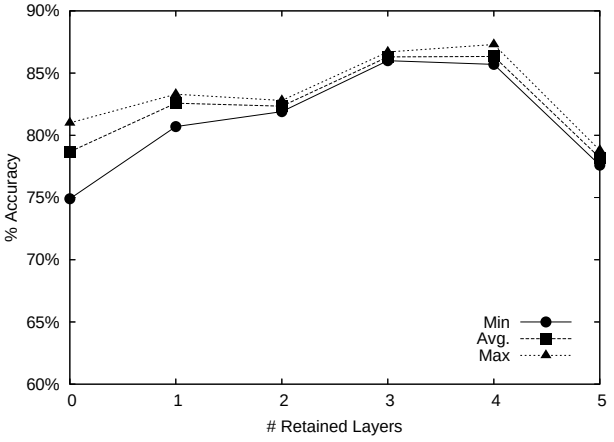

Figure 5: Comparison of learning curves showing accuracy vs. number of retained levels for 40 samples per class in the training set. Curves show minimum accuracy, average accuracy and maximum accuracy obtained over 5 trials on a testing set with 1,000 character samples

Layer	*Free Parameters*
C1	156
S2	12
C3	1,516
S4	32
C5	48,120
F6	2,420
Total	52,256

Table 1: Number of free paramters at each layer of our net

available to it. When the top 2 levels are being retrained the net has 96.7% of its free parameters available for training.

Discussion & Analysis

The main process being shown in Fig. 2 is a trade-off between increased net capacity, as reflected by fewer retained levels/free parameters, and increased knowledge transfer as reflected by more retained levels. As more information is available to the net through increased training set size, the importance of transferred knowledge decreases and having sufficient capacity to learn the details of the new classes increases.

By observing the change in shape of the learning curves, it is possible to get a qualitative feel of this effect. The shapes of the learning curves for 1 sample/class, 10 samples/class and 40 samples/class may be seen in greater detail in Figs. 3-5. The minimum, average and maximum accuracy obtained over each 5 trial run is shown to illustrate the relatively slight variance that was observed. These figures highlight the way in which each layer contributes to the transfer of knowledge from the source net. Furthermore, they emphasize the changing shape of the learning curve as the increase in training set size makes net capacity more important relative to transferred knowledge. This, however, may be taken with equal justification to be an indicator of the quality of the knowledge transferred. It seems likely that if the source net had learned either a larger set of classes, then the benefit of knowledge transfer would be greater and persist for even larger training sets. Perhaps a different set of classes, which in some sense spanned the set of classes better, would also give improved results.

It is interesting to observe how much of an advantage is obtained merely by retaining just the bottom four levels. Although these levels contain only 3.3% of the weights used by the net, their transfer leads to marked improvements in the accuracy of the net.

One may observe in Fig. 2, that when levels C1-C5 are retained, the net doesn't seem to have sufficient capacity to learn appreciably more information than is contained in about 10 samples. This implies that when attempting knowledge transfer between two DCNN's, slightly more flexibility in choosing which weights should be retrained could be beneficial. For instance, perhaps, one could retrain only some of the feature maps at a given level, rather than all or none. This would enable us to have a partially transferred layer

between the fully transferred and fully trained layers, which could help fine tune the balance between transferred knowledge and net capacity.

Lastly, one might also consider letting retraining take place with the transferred feature maps. This would undoubtedly give better results than were obtained, however, part of what we wanted to see was how much could be learned, without forgetting previously acquired concepts. One can now envision a particular sub-net being shared among several nets, each of which has been trained for different tasks.

Conclusions & Future Work

Our results show that for small training sets there is a clear advantage to favoring knowledge transfer over net capacity in both accuracy of learning and in effort required to learn.

What remains to be investigated is how quickly this trade-off changes as the number of classes that contribute to knowledge transfer increases. Bartlett & Baxter's results (Bartlett & Baxter 1998) strongly suggest that ultimately knowledge transfer will achieve accuracy comparable to the best achievable for full capacity nets with large training sets.

We plan to investigate methods of optimizing this trade-off, by allowing some feature maps at a given level to retrain. It should be possible adapt saliency, as in Optimal Brain Damage (LeCun *et al.* 1990), or to adapt mutual information, as in DBT (Pratt 1993), to determine which feature maps should be transferred and which should be retrained.

Finally, we will investigate techniques to select an optimal set of classes for knowledge transfer.

References

Bartlett, P., and Baxter, J. 1998. The canonical distortion measure in feature space and 1-nn classification. *Advances in Neural Information Processing Systems* 10:245–251.

Baxter, J. 2000. A model of inductive bias. *Journal of Artificial Intelligence Research* 12:149–198.

Bishop, C. 1995. *Neural Networks for Pattern Recognition*. Oxford University Press.

Caruana, R. 1997. Multi-task learning. *CMU Technical Reports: CMU-CS-97-203*.

Chopra, S.; Hadsell, R.; and LeCun, Y. 2005. Learning a similarity metric discriminatively, with application to face verification. In *Proc. of Computer Vision and Pattern Recognition Conference*. IEEE Press.

Fahlman, S., and Lebiere, C. 1990. The cascade-correlation learning architecture. In Touretzky, D. S., ed., *Advances in Neural Information Processing Systems*, volume 2, 524–532. Denver CO: Morgan Kaufmann, San Mateo.

LeCun, Y.; Denker, J.; Solla, S.; Howard, R.; and Jackel, D. 1990. Optimal brain damage. In Touretzky, D., ed., *Advances in Neural Information Processing Systems 2 (Neural Information Processing Systems*89)*. Denver, CO: Morgan Kaufman.

LeCun, Y.; Haffner, P.; Bottou, L.; and Bengio, Y. 1999. Object recognition with gradient-based learning. In Forsyth, D., ed., *Feature Grouping*. Springer.

Pratt, L. 1993. Discriminability-based transfer between neural networks. *Advances in Neural Information Processing* 5:204–211.

Schultz, T., and Rivest, F. 2000. Knowledge-based cascade corellation. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks 2000*, V641–V646.

Simard, P.; LeCun, Y.; Denker, J.; and Victorri, B. 2001. Transformation invariance in pattern recognition – tangent distance and tangent propagation. *International Journal of Imaging Systems and Technology* 11(3).

Simard, P.; Steinkraus, D.; and Platt, J.C., t. .

Thrun, S. 1995. Lifelong learning: A case study. *CMU Technical Reports: CMU-CS-95-208*.