A Self-Learning Visual Pattern Explorer and Recognizer using a Higher Order Neural Network

Günter Linhart *Georg Dorffner*

Austrian Research Institute for Artificial Intelligence
email: guenter@ai.univie.ac.at, georg@ai.univie.ac.at

Abstract

Higher order neural networks, although known for their power, have not been acknowledged very much in literature, mainly due to their apparent computational complexity. In this paper a proposal ([Reid *et al.* 1989]) to improve the efficiency of such networks is taken and built into a pattern recognition system that autonomously learns to categorize and *recognize patterns independent from their position* in an input image. It does this by combining higher order with first order networks and the mechanisms known from ART. Its recognition is based on a 16 x 16 pixel input which contains a section of the image found by a separate centering mechanism. It can be shown, that with this system position invariant recognition can be implemented efficiently, while combining all the advantages of the employed sub-systems.

Introduction

A major problem in pattern recognition with neural networks is the dependence of recognition on changes of position and size of the pattern in the input layer. A variety of solutions exist to solve these invariance problems, but they all rely on traditional preprocessing techniques (such as fourier transformation, feature detectors, or the like) or on the use of multilayer networks with considerable complexities of implementing and learning (e.g. the Neocognitron). In this paper, a completely different method is introduced that makes use of specialized higher order networks.

Already in the 1960s the possibilities of higher order neural networks (HONNs) were pointed out ([Minsky & Papert 1969]). Due to the combinatorial explosion with respect to the necessary weights and computations this approach has not been considered practically useful. On the other hand, a decisive advantage of higher order networks is their computational power without hidden layers, comparable to that of multi-layer perceptrons ([Pao & Beer 1988], [Reid *et al.* 1989]). Therefore, simple and fast learning rules can be used (e.g. Hebbian or delta rules), while the latter need complex training mechanisms (e.g. backpropagation) with their known problems (many iterations, local minima). Further advantages are their provability of convergence in recursive applications ([Lee *et al.* 1985]) and in many cases the easy interpretability of the resulting activations in the network.

The above-mentioned large number of connections can be avoided if the specific use of the network is taken into consideration during its design. Through appropriate reduction of connections to a structure necessary for a specific application the behavior of the network can be tuned to the requirements, at the same time increasing its efficency.

Theoretical Background

In a *first order neural network (FONN)* the net input is computed according to

$$net_i = \sum_{j=0}^{N-1} x_j w_{i,j}. \qquad (1)$$

If weighted products of different pre-synaptic activations are included into this computation, the network is said to be of order greater than one. The degree of order corresponds to the largest number of factors in any product. In Fig. 1 a simple first order network is shown compared to a simple network of order two, each with four input units and one output unit. In the second order network all connections necessary for computing the products are depicted, while the "direct" first order connections - which would make it the most general network of order two -

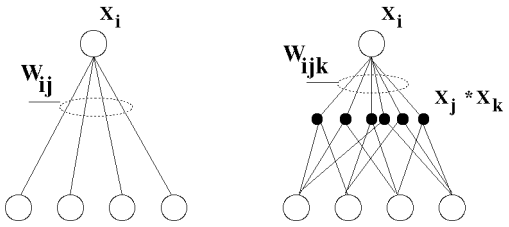

Fig. 1: **A simple first order network (left) in comparison to a simple network of order two**

are not shown for the sake of clarity. Extending formula (1) to the most general case of a higher order network one gets

$$net_i = \textbf{Fehler!}$$

where x_j and x_k are the activations of unit i and k, respectively, $w_{i,j}$ is the weight between the j-th unit in the input layer and the i-th unit in the output. Accordingly, $w_{i,j,k}$ is the weight between the i-th output unit and the product of the j-th and the k-th input unit activation. In the third order term, $w_{i,j,k,l}$ is the weight of the product $x_j x_k x_l$ to unit i, and so on.

If direct connections between the input and subsequent layers are omitted, formula (2) for a second order network reduces to the second term. Now the desired invariances can be built in by designing the learning mechanism such that all weights of equal relevance to a particular problem are adapted equally ([Reid *et al.* 1989]). This way, all connections with equal weights can be combined into one, and the major disadvantage of HONNs - the combinatorial explosion of the number of weights - can be avoided.

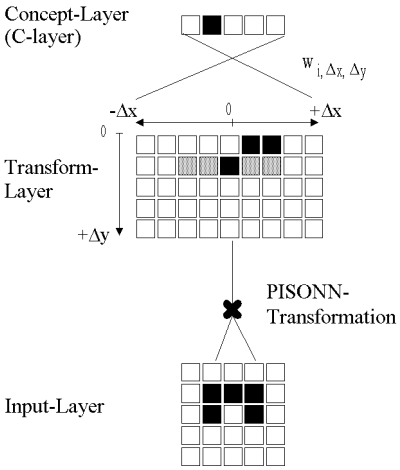

Fig. 2: **A position invariant second order neural network**

As proposed in [Reid *et al.* 1989] position invariance can simply be achieved through using the same weights for each combination of two units with a fixed relative distance. Thus a special form of second order networks is derived, which is henceforth referred to as *PISONN (position invariant second order neural net)*. In the following formalisms, the symbolic macros XPOS() and YPOS() are used for simplicity, which denote the x- and y-positions of a unit u_j or u_k respectively. The net input of an output unit in a PISONN is computed as

$$net_i = \sum_{j=0}^{N-2} \sum_{k=j+1}^{N-1} x_j x_k w_{i,j,k} \qquad (3)$$

with $w_{i,j,k} = w_{i,\Delta x,\Delta y}$ *for all j,k*
and $\Delta x = XPOS(x_k) - XPOS(x_j)$, $\Delta y = YPOS(x_k) - YPOS(x_j)$.

Through iteration over k = j+1 .. N-1 it is ensured that each pair of units enters the computation only once. For an efficient implementation the products of two units are summed up in an extra hidden unit layer (called *transform-layer*), whose activations are subsequently multiplied with the (now small number of) weights (Fig. 2). The activations in those hidden units are computed as the sum of all products of two units with exactly the difference in the x- and y-coordinate the hidden unit corresponds to. This way, the hidden activations have to be computed only once per input pattern. During training only the weights between hidden and output layer are adapted, so that the amount of computation is merely proportional to the size of the input layer - with the exception of the initial transformation for each new pattern.

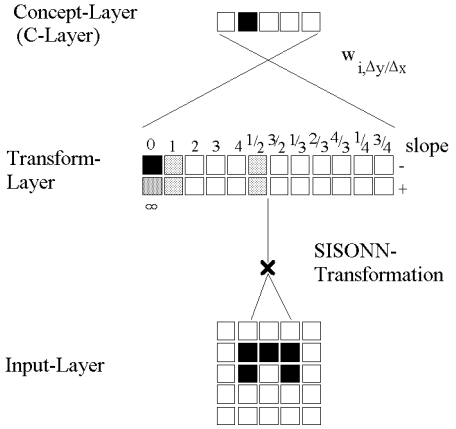

Fig. 3: **A size invariant second order neural network**

If in addition to position invariance also size invariance is desired, equal weights are used for each two points in the input layer that lie on a straight line with a fixed angle. Here, too, a special form of second order networks is derived. It will be referred to as *SISONN (size invariant second order neural net)* and has a net input of

$$net_i = \sum_{j=0}^{N-2} \sum_{k=j+1}^{N-1} x_j x_k w_{i,j,k} \qquad (4)$$

with $w_{i,j,k} = w_{i,\Delta y/\Delta x}$ *for all j,k*
and $\Delta x = XPOS(x_j) - XPOS(x_k)$, $\Delta y = YPOS(x_j) - YPOS(x_k)$.

In analogy to the PISONN, by insertion of a hidden (transform) layer the necessary multiplications are reduced to an operation to be done only once (Fig. 3). Each hidden unit now corresponds to a specific angle of the line containing the two points represented by two input units. As this network shows some kind of position invariance in addition to the size invariance, this type of network was chosen for the model introduced here.

The goal of the introduced model is to classify different objects in an image at the input into new or existing categories. For the categorizaton a special purpose layer - called a c*oncept-(C-)layer* was used that contains inhibitory intra-layer connections, which are adapted, together with the connections leading to the C-layer, according to a "winner-take-more" (WTM) rule ([Dorffner 1989]). This rule amplifies strongly activated units and suppresses the others, while preserving the basic distributed structure of the activations, in contrast to the brittle winner-take-all (WTA) rule. In other words, a weakly distributed pattern can remain active while favoring one strongly activated unit. This type of update contains more information than a unitized pattern, which is appropriate for many tasks described elsewhere (e.g. [Dorffner 1989]).

Model Architecture

The overall system essentially consists of a modified ART model ([Carpenter & Grossberg 1987]). In addition to a feedback loop for resonance of first order categories in the input, it contains a similar second order loop. Furthermore, an exploration component is added which automatically and sequentially generates patterns to be learned. This component consists of a centering mechanism which focusses on a 16 x 16 pixel part of the 100 x 100 pixel input image. Fig. 4 schematically depicts the overall architecture. The input layer (*Vision*) contains the current section of the larger image (*Environment*). The exploration component choosing a particular section is (similar to the model of reinforcement learning in [Mannes 1990]) controlled by a motor layer (*Motor*), which consists of two trinary units corresponding to two directions of movement (X and Y), and a random component (*Random*). In addition - whenever necessary - a pattern in the transform layer (*Transform*) on the basis of the SISONN described above is activated. Categorization is done in the C-layer which, together with a masking layer (*Mask*), forms the conceptualization component. Two additional layers - *Imagine Vision* and *Imagine Transform* - are introduced for implementing class prototypes and the comparison with the corresponding vision and transform layers.

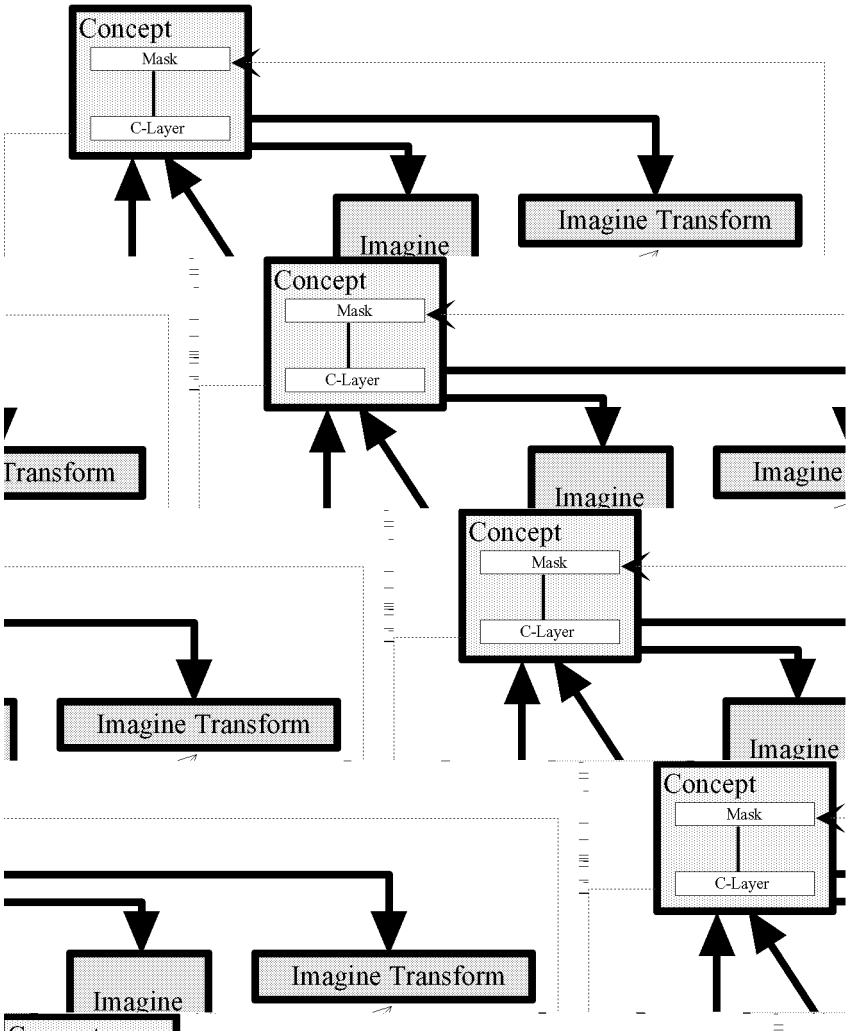

Fig. 4: **Architecture of the autonomous learning model (schematically)**

Operation and Learning

The introduced model, learning autonomously, combines a variety of advantages of the different employed network approaches. The first order network permits all categorisation tasks inherent to the ART model, whenever the position of the pattern in the input layer is fixed. In addition, the second order network permits the model to recognize learned patterns also on arbitrary different positions in the input. To avoid hypersensitivity of the units in the C-layer the recognition of patterns in shifted positions (through second order resonance) does not

lead to fullfledged learning. Only the weights between the transform and the C-layer are adapted, so that after some training a second order update leads to the activation of the correct unit in the C-layer.

A cycle of recognition starts with the update from the vision to the C-layer. The winner of the C-layer is selected (through WTA) and activation propagated to the imagine vision layer. For known patterns a prototype is activated, which is compared to the pattern in the vision layer. If the similarity between the two patterns is above the value of the vigilance parameter ρ_1 the category is accepted. If it is not, the category is not immediately rejected (as in regular ART). Via the SISONN a pattern in the transform layer is activated and compared to the pattern in the imagine transform layer, after propagation via the C-layer. A second vigilance parameter ρ_2 decides upon acceptance or rejection of the category. This way, different patterns on the same position are categorized through the first order network, and similar patterns of different positions through the SISONN. The two resonance mechanism harmonically complement each other by softening the usually strict border between resonance and mismatch (*fuzzy match*).

Automatic centering is another part of the system, being relatively independent from the rest. With the help of this mechanism the model is able to autonomously "pick out" sections of the image in order to categorize them. It takes care of large position shifts, while the SISONN is responsible for unavoidable local shifts within the vision layer. First the position of the window is chosen randomly. As soon as a part of a pattern is located, the centering mechanism (via the motor layer) shifts the window such that the whole pattern is visible, provided its size is not too large. After catgeorization of this visual pattern another random jump is initiated starting the cycle anew. The exploration phase is ended when either all available category units are committed, or when no new categories are discovered and all known ones have reached a certain degree of compression (the so-called ID-state, see below).

Similar to [Grossberg 1988], activations in the C-layer are computed according to a shunting equation (or interactive activation update - [McClelland & Rumelhart 1981]). The update in the imagine layers for visualization of the respective patterns is done with a simple linear transfer function. For training the C-layer, the learning rules introduced in [Dorffner 1989] and [Dorffner 1992] are used. This "winner-take-more" mechanism not only adapts the weights to the most-activated unit (winner), but to other highly activated ones also. A special parameter is introduced - the "degree of winning" - which guides weight adaptation such that similarities in the responses between classes are decreased, while similarities within classes have a chance to remain active. With this parameter training only gradually compresses the responses toward a unitized pattern (where only the winner remains active). In the following learning rule highly active units are distinguished from weakly active ones, expressed by the predicates ON(i) and OFF(i).

$$\Delta w_{ij} = \eta_1 x_i x_j, \quad \text{if ON(i) \& ON(j) \& } win_i = N\text{-}1 \text{ } (x_j \text{ is the winner}) \qquad (5)\ (a)$$

$$= \eta_2 x_i \frac{win_i - best_i}{N\text{-}1}, \quad \text{if ON(i) \& ON(j) \& } win_i < N\text{-}1 \text{ } (x_j \text{ is not the winner}) \qquad (b)$$

$$= -\eta_2 x_j, \quad \text{if OFF(i) \& ON(j)} \qquad (c)$$

$$= 0, \quad \text{otherwise} \qquad (d)$$

where win_i is the degree of winning of unit i (basically the relative rank of the unit in the distributed pattern), and $best_i$ the stored best value of win_i the unit has ever reached. N is the size of the C-layer. The *intra-layer connections* are adapted according to a learning rule similar to one proposed in [Marshall 1989], except that no weight increase is done in the reverse direction:

$$\Delta w_{ij}^{inh} = -\eta_1 x_i x_j, \quad \text{if } x_i > x_j \qquad (6)$$

$$= 0 \quad \text{otherwise}$$

Typical values for the learning rates are $\eta_1 = 0.0005$ and $\eta_2 = 0.05$. As thresholds, the values $\Theta_1 = 0.5$, $\Theta_2 = 0.3$ and $\Theta_{ON} = 0.1$ were used. In addition, the weights on the connections leading to the C-layer are bounded to stay in the interval [0 .. 1]. The training cycle is repeated until a state of high compression (the said *identifiable (ID-) state*) is reached. It is defined as the state in which the winner is considerable higher (threshold Θ_1) than the average of the other units, and also considerable higher (threshold Θ_2) than the secondmost active unit (to avoid two strongly active units). To train the connections between the C-layer and the imagine layers a simple delta rule was used, which adapts the weights from the winner in proportion to the difference between current patterns in the vision (or transform) and the corresponding imagine layers. The learning rate ε was chosen for each particular task, usually around 0.1. If a new category is detected, a rate of $\varepsilon = 1.0$ is taken, so that the prototype in the imagine layer is set equal to the current input pattern. A few additional provisions - such as permitting adaptation

only in the case of recognition in the first order network, or learning only until an ID-state is reached - leads to a stable prototype even for unusual sequences of training patterns.

The *comparison function* used for matching the patterns between the vision (respectively transform) layer and the corresponding imagine layer is the following (see [Linhart 1991] for more details):

$$r_1 = 1 - \frac{\sum_i |x^V_i - x^{IV}_i|}{\sum_i x^V_i + x^{IV}_i} \qquad (7)$$

with x^V_i *resp.* x^{IV}_i *... Unit i in vision layer resp. imagine vision layer.*

This function computes a relative measure of similarity between two patterns of activations in the interval [0 .. 1]. Two vigilance parameters ρ_1 and ρ_2 (described above) were used as threshold to decide upon resonance or mismatch, which can be set by the user.

Results

To test the *position invariant recognition* a set of five different patterns was used, which were permitted to occur on three different positions in the V-layer. Due to a relatively high learning rate, ID-state was reached after only 100 pattern presentations. Through cutting off learning at this point, no oversensitivity of the C-units can be observed. This is one of the major advantages of the used network architecture. The automatic comparison with the prototype makes the categorization mainly dependent on the two vigilance parameters, and a high learning rate cannot lead to incorrect categorizations. The five categories were learned almost independently from ρ_1, whereas varying ρ_2 lead to categorization into any number between one ($\rho_2 = 0.3$) and five classes.

The *size invariant recognition* works without problems only for simple line drawings. With more complex images the products in the transform layer lead to strong overlaps and thus sometimes to wrong classifications. Still, a considerable number of complex patterns with small distortions were recognized, even though the unit overlap was small (leaving no chance to the first order network for correct recognition). Another interesting property of the SISONN is its invariance towards patterns rotated by 180 degrees. This results from the commutative computation of the products in (2).

To test the categorization abilities in connection with the exploration system, several handwritten digits were put together on a 100 x 100 pixel input image. Each digit was included several times in different shapes. The task was to detect the digits, to classify them together into categories, and to recognize them independent from their position. Depending on the vigilance parameter this task was correctly learned with up to 100 % correct recognition (Fig. 5). In the case of two shapes of the same digits that were too dissimilar, however, misclassifications could be observed. A series of additional tests, even using patterns connected to each other, all revealed the central tendency of the network model - position invariance was perfect, size and distortion invariance only limited, but still better than with the comparable first order network.

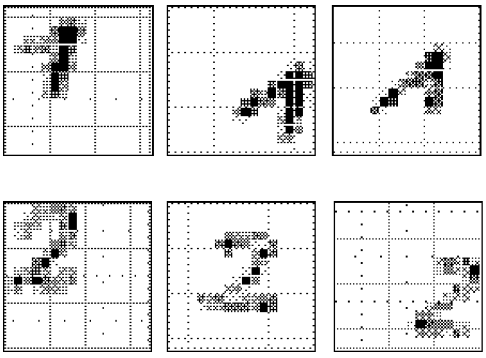

Fig. 5: **Examples of correctly recognized numbers which are categorized in two classes "1" and "2"**

Discussion

This work has shown, in confirmation of the work in [Reid *et al.* 1989], that higher order networks can be implemented and trained rather efficiently, if assumptions specific to a certain application are included in the network design. In the case of visual pattern recogntion, as described in this paper, position invariant recognition can easily be achieved. The storage of invariant information is enabled through the redundancy of the theoretically large number of weights. Due to efficient implementation with the help of a hidden layer the amount of computation for training is approximately the same as for a two-layer network of first order. Owing to the increased pattern similarities in the hidden unit activations after transformation, this type of HONN by itself does not lend itself to unsupervized learning tasks. In combination with the ART control mechanism, described above,

this disadvantage can be partially avoided. The integration of a first order network and the SISONN plus the ART mechanisms helps exploit the advantages of both networks without their drawbacks (position dependency of the FONN, bad categorization of the HONN). In addition, with the help of the vigilance parameters, the properties of categorization can be adapted to the user's needs, an important asset for practical applications.
In connection with the exploration system the model introduced in this paper is able to autonomously discover a number of small patterns in a large image, to categorize and to recognize them. In the implementation using a neural network toolkit, developed at our Institute, learning of about 20 patterns on a 100 x 100 pixel images takes only a few minutes on a 486 AT. After such training it is able to recognize each object completely independent of its position, provided it is small enough to fit in the vision layer. Even after cutting off 10 - 20 % of the pattern on one of its edges a good recognition rate was achieved. If the patterns are varied in size, the recognition rate, dependent on the structure of the patterns and their similarity to other patterns, is modest but still superior to the results with a mere first order network. Further investigations will be into the applications of HONNs to other pattern recognition domains, such as speech recognition, sequential pattern recognition, and rotation invariance. Furthermore, research is being undertaken to extend the C-layers to include information about (geometrical or other) relations between patterns. Finally, it is hoped to improve the abilities of HONNs through the introduction of habituation mechanisms on hidden units frequently activated, which are expected to reduce the large pattern similarities in the transform layers.

References

[Carpenter & Grossberg 1987] Carpenter G.A., Grossberg S. (1987): A massively parallel architecture for a self-organizing neural pattern recognition machine, in Computer Vision, Graphics, an Image Processing, 37, p.54-115.

[Dorffner 1989] Dorffner G. (1989): A Sub-Symbolic Connectionist Model of Basic Language Functions, Indiana Univerity, Computer Science Dept., Dissertation.

[Dorffner 1992] Dorffner G. (submitted): "Winner-Take-More" - A Mechanism for Soft Competitive Learning, Austrian Research Institute for Artificial Intelligence; submitted for publication.

[Grossberg 1988] Grossberg S. (1988): Competitive Learning: From interactive activation to adaptive resonance, in: Neural Networks and Natural Intelligence, Chapter 5, A Bradford Book, MIT Press, Cambridge, MA.

[Lee *et al.* 1985] Lee Y.C., Doolen G., Chen H.H., Sun G.Z., Maxwell T., Lee H.Y., Giles C.L. (1985): Proceedings Evol.Games & Learn, May, Los Alamos.

[Linhart 1991] Linhart G. (1991): Über die Eignung von neuronalen Netzwerken höherer Ordnung für die visuelle Bildverarbeitung, master's thesis, Dept. of Med. Cybernetics and AI, University of Vienna.

[Mannes 1990] Mannes C. (1990): Learning Sensory-Motor Coordination by Experimentation and Reinforcement Learning, in Dorffner G.(ed.), Konnektionismus in Artificial Intelligence und Kognitionsforschung, Springer, Berlin, p.95-102.

[Marshall 1989] Marshall J.A. (1989): Self-Organizing Neural Network Architectures for Computing Visual Depth from Motion Parallax, in IEEE International Conference On Neural Networks, Washington D.C., IEEE, Volume II, p.227-234.

[McClelland & Rumelhart 1981] McClelland J.L., Rumelhart D.E. (1981): An Interactive Activation Model of Context Effects in Letter Perception: Part 1. An Account of Basic Findings, Psychological Review, Vol. 88, p.375-407.

[Minsky & Papert 1969] Minski M.L., Papert S. (1969): Perceptrons, MIT Press, Cambridge.

[Pao & Beer 1988] Pao Y.H., Beer R.D. (1988): The functional link net: a unifying network architecture incorporating higher order effects, International Neural Network Society First Annual Meeting, Bostan, MA.

[Reid *et al.* 1989] Reid M.B., Spirkovska L., Ochoa E. (1989): Rapid Training of Higher-Order Neural Networks for Invariant Pattern Recognition, IJCNN.

[Rumelhart *et al.* 1986] Rumelhart D.E., McClelland J.L., et al. (1986): Parallel Distributed Processing, Explorations in the Microstructure of Cognition, Volume 1: Foundations, MIT Press, Cambridge, Massachusetts.