Energy-Based Models in Document Recognition and Computer Vision.

Yann LeCun, Sumit Chopra, Marc'Aurelio Ranzato, and Fu-Jie Huang
The Courant Institute of Mathematical Sciences
New York University
http://yann.lecun.com

Abstract

The Machine Learning and Pattern Recognition communities are facing two challenges: solving the normalization problem*, and solving the* deep learning problem*.*

The normalization problem is related to the difficulty of training probabilistic models over large spaces while keeping them properly normalized. In recent years, the ML and Natural Language communities have devoted considerable efforts to circumventing this problem by developing "un-normalized" learning models for tasks in which the output is highly structured (e.g. English sentences). This class of models was in fact originally developed during the 90's in the handwriting recognition community, and includes Graph Transformer Networks, Conditional Random Fields, Hidden Markov SVMs, and Maximum Margin Markov Networks. We describe these models within the unifying framework of "Energy-Based Models" (EBM).

The Deep Learning Problem is related to the issue of training all the levels of a recognition system (e.g. segmentation, feature extraction, recognition, etc) in an integrated fashion. We first consider "traditional" methods for deep learning, such as convolutional networks and back-propagation, and show that, although they produce very low error rates for handwriting and object recognition, they require many training samples. We show that using unsupervised learning to initialize the layers of a deep network dramatically reduces the required number of training samples, particularly for such tasks as the recognition of everyday objects at the category level.

1. Two Challenges in Machine Learning

It may come to a surprise to many members of the ICDAR community that some of the recent advances in Machine Learning have their root in the document recognition literature. What used to be called statistical pattern recognition, and has become the core of the expanding field of machine learning (ML), has always been a key component of document recognition systems. But for many other areas of computer and information science, such as natural language processing, computer vision, and robotics, the widespread adoption of ML methods is relatively recent.

Interestingly, there are at least two important classes of ML methods that were first developped in the context of document image recognition: 1. discriminative training methods for "structured output" problems, particularly the non-probabilistic kind; 2. learning algorithms and architectures that can simultaneously train the classifier and the feature extractor in an integrated fashion. These two classes of methods address two recurring problems in machine learning: the *normalization problem*, and the *deep learning problem*.

The normalization problem arises in the context of document recognition when simultaneously training a segmenter, a recognizer, and a language model discriminatively. This capability is essential for training handwriting recognition systems at the word or sentence level, without requiring prior manual segmentation of the characters. The traditional solution has long been to use a combination or Neural Networks and Hidden Markov Models (HMM) [16, 6]. However, the internal normalization of HMMs often leads to inconsistent decisions, a problem first pointed out by Bottou [5], and later named the "label bias problem" by Lafferty et al. [15]. Early solutions involved "late normalization schemes" which, instead of manipulating normalized probabilities in the HMM, merely manipulate costs (which may be interpreted as un-normalized negative log probabilities) [8]. This line of work eventually led to the Graph Transformer Network model for document recognition which uses either a negative log-likelihood loss function or a Perceptron-like loss function [18]. In recent years, several authors have also proposed similar "un-normalized" models for discriminative sequence labeling, notably conditional random fields [15], perceptron-like models [7], support vector Markov models [1], and maximum margin Markov networks [28]. This has spurred a considerable amount of interest for these in natural language parsing, machine translation, bioinformatics, and computer vision. The common feature of all these models is the manipulation of *energies* instead of normalized probabilities. They can all be described using the unifying concept of *energy-based model* (EBM), which is briefly introduced in subsequent sections. More details can be found in a recent tutorial paper [19].

The deep learning problem arises when learning highly complex tasks, such as invariant shape recognition [4]. A time-honored approach to pattern recognition is to use a manually designed feature extractor, followed by a generic trainable classifier. However, attempts to solve tasks of increasing complexity and diversity have lead several researchers to seek methods that can learn the feature extractor and the classifier in an integrated fashion. The motivation is to produce methods that can be applied to a wide variety of problems, without requiring a large amount of skilled labor for the design of a task-specific preprocessor. One approach is to dispense with the feature extractor altogether, and simply apply a generic classification method, such as a support vector machines (SVM) or a two-layer neural net to raw pixels. Unfortunately, SVMs and simple neural nets are *shallow architectures* in which the entire classification function is implemented with only two stages of non-linearities. There is theoretical and empirical evidence that such shallow architectures are extremely limited when it comes to learning high-level tasks involving complex invariances [4, 14]. A more appropriate approach is to use a *deep architecture*, composed of multiple successive layers that can perform feature extraction and classification (and sometimes segmentation as well) in an integrated fashion. Well-designed deep architectures can be trained with gradient descent (using back-propagation) in supervised mode. A a notable example of such end-to-end supervised learning system is the Convolutional networks [17, 18]. Convolutional networks typically comprise 4 to 8 layers, each of which can be seen as a bank of convolutional filters followed by a point-wise non-linearity. Each convolutional layer is often followed by feature-pooling layer that reduces the resolution through a local max or averaging operation. Among purely learning-based approach applied to raw pixels, they hold the record on the MNIST handwritten numeral dataset, with 0.4% error rate [27]. They have also been applied successfully to a variety of Computer Vision and Pattern Recognition tasks, such as face detection [10, 24], generic object recognition [20, 14], and visual navigation for mobile robots [21]. However, due to the large number of free parameters involved, such purely supervised, end-to-end training requires a fair amount of training samples to reach good accuracy. Moreover, while back-propagation works reliably with convolutional nets, the same cannot be said when it is applied to fully-connected neural nets with many layers.

While the general problem of training deep architectures is still considered largely unsolved, a new promising approach has started to emerge in the last two years called *Deep Belief Networks* (DBN) [12, 13, 3, 23]. The main idea of DBN training is to initially train each layer one-by-one in an unsupervised fashion. Each layer is composed of an encoder, which maps inputs to outputs (or feature vectors), and a decoder that stochastically reconstructs the input from the output. The layer is trained to model the distribution of input vectors. Each layer is fed with the output of the previous layer. Once the unsupervised learning is complete, the chain of encoders forms a feed-forward network that extract representations of increasingly high level. This feed-forward network can then be fine-tuned with supervised back-propagation. The method produces excellent, record-braking results on MNIST numerals [12, 23]. Futhermore, it allows a drastic reduction of the required number of labeled examples, which is particularly important for computer vision tasks in which the number of training samples is as low [25] (as low as 30 samples per category for the popular Caltech-101 object recognition benchmark).

The following sections describe the Energy-Based Models approach and the Deep Belief Network approach in more details.

2. Energy-Based Models

This section gives a brief introduction to energy-based learning. A more detailed treatment can be found in [19]. Recognition systems capture the dependencies between a set of observed variables X, such as the pixels of an image, and a set of *answer variables* Y to be predicted (e.g. the words in a text image, or the catgories of the objects in a natural image). An EBM takes in all the variables (observed or unobserved) as inputs, and produces a scalar *energy* $E(Y, X)$ which measures the "compatibility" between the values of the variables. The *inference process* (finding the best answer for a given input) consists in finding the values of Y that is most compatible with the observed input, i.e., the ones that minimizes the energy:

$$Y^* = \mathrm{argmin}_{y \in \mathcal{Y}} E(y, X), \tag{1}$$

where $\mathcal{Y}$ is a suitably defined domain for Y. One can transform an EBM into a probabilistic model through the Gibbs distribution

$$P(Y|X) = \frac{e^{-\beta E(Y,X)}}{\int_{y \in \mathcal{Y}} e^{-\beta E(y,X)}}, \tag{2}$$

where β is an arbitrary positive constant. This transformation requires that the integral in the denominator (the *partition function* converges. The EBM inference through energy minimization can be seen as a maximum a posteriori estimation of Y.

In many applications, some the relevant variables are neither inputs nor answers. They are referred to as *latent variables* and denoted Z. For example, in handwriting recognition, the correct segmentation of the sentence into words and characters is never given to us, but it must be inferred in order to recognize the sentence. The recognition process can be seen as simultaneously searching for the best answer and the best segmentation. We can view this as simultaneously minimizing an energy function over Y and Z:

$$Y^* = \mathrm{argmin}_{y \in \mathcal{Y}} \min_{z \in \mathcal{Z}} E(z, y, X), \tag{3}$$

This comes down to redefining the energy as $E(Y, X) = \min_{z \in \mathcal{Z}} E(z, y, X)$, and applying equation 1. In a probabilistic framework, latent variables must be marginalized over, instead of minimized over. This comes down to redefining the energy as $E(Y, X) = -1/\beta \log \int_{z \in \mathcal{Z}} e^{-\beta E(z,y,X)}$, and applying equation 2.

A typical EBM-type architecture for handwriting recognition is represented in figure 1.

2.1. Training an EBM

Since EBM inference produces values of Y with the smallest energy for a given X, training an EBM will consist in finding an energy function that produce a lower energy

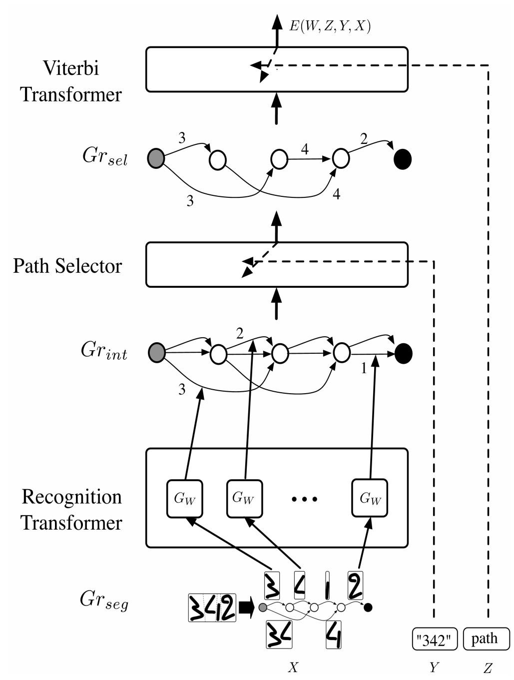

Figure 1. *The architecture of an Energy-Based Model for handwriting recognition. The input X is an image of a word, the segmentation of which is unknown a priori. Y is the label sequence, and Z is the* latent segmentation variable *which encodes possible ways in which the input image can be segmented into characters. In many handwriting systems, a particular segmentation and interpretation is represented as a path in a graph. The inference process simultaneously mininizes $E(Z, Y, X)$ with respect to Y and Z, which is performed by running a shortest-path algorithm on the graph.*

for the desired value(s) of Y than for all other values. First we define a family of energy functions indexed by a parameter W, among which our learning algorithm will find the best one, $\mathcal{E} = \{E(W, Y, X) : W \in \mathcal{W}\}$. Given a set of labeled training samples $\mathcal{S} = \{(X^i, Y^i) : i = 1 \dots P\}$, we must design a *loss functional* $\mathcal{L}(E(W, ., .), \mathcal{S})$, whose role is to indicate how well (or how badly) an energy function $E(W, ., .)$ performs on the training set $\mathcal{S}$. To simplify, we use the notation $\mathcal{L}(E(W, ., .), \mathcal{S}) = \mathcal{L}(W, \mathcal{S})$ Training the EBM comes down to finding the W^* that minimizes the loss. The loss is defined as an average over the training set of a *per-sample loss*:

$$\mathcal{L}(W, \mathcal{S}) = \frac{1}{P} \sum_{i=1}^{P} L(Y^i, E(W, \mathcal{Y}, X^i)) + R(W), \quad (4)$$

where $R(W)$ is a regularization term that embeds our prior knowledge about which energy functions are preferable to others. the per-sample loss must be designed in such a way that its minimization will have the effect of making the energy of the correct answer $E(W, Y^i, X^i)$ lower than the energy of any other answer $E(W, y, X^i)$, $y \neq Y^i$. Several types of loss functions have been proposed for this purpose, going back to the early days of discriminative training for speech recognition. They include the negative log-likelihood loss [2], empirical error [22], and LVQ2 [9]. For handwriting recognition, the negative log-likelihood [16, 18], and the perceptron criterion [18] have been used for many years.

The most common loss function is the negative log-likelihood:

$$L_{\text{nll}}(W, Y^i, X^i) = E(W, Y^i, X^i) + \frac{1}{\beta} \log \int_y e^{-\beta E(W, y, X^i)}$$

For a single training sample (Y^i, X^i), this is simply the negative log of $P(Y^i | X^i)$ as given by equation 2. Minimizing this loss can be performed with a stochastic gradient-based procedure:

$$W \leftarrow W - \eta \left(\frac{\partial E(W, Y^i, X^i)}{\partial W} - \int_y P(y | X^i) \frac{\partial E(W, y, X^i)}{\partial W} \right)$$

Where η is a suitably chosen step size (a positive, semi-definite matrix). The first term "pushes down" on $E(W, Y^i, X^i)$ and the second term "pulls up" (with the same total force) on the energies of all values of Y. Unfortunately, *evaluating the integral in the second term may be intractable*. This is the main manifestation of the *normalization problem* mentioned in the introduction. The problem is commonly approached through Monte-Carlo sampling or variational methods.

One the main motivations behind the EBM approach is to circumvent this problem by devising other loss functions that do not involve intractable integrals. A popular loss function is the so-called perceptron loss [18, 7]:

$$L_{\text{ptron}}(W, Y^i, X^i) = E(W, Y^i, X^i) - \min_y E(W, y, X^i)$$

This loss merely pushes up on the energy of the best answer produced by the system. It is the limit of the negative log-likelihood loss for large values of β. Unfortunately, while the perceptron loss works well in most cases, there is no guarantee that the energy of the desired answer will be *strictly* lower than the others. This loss could result in a completely flat energy surface in which every answers have the same energy. To fix this problem, several authors have proposed loss functions with a *margin*. First, we must define the *most offending incorrect answer*, denoted $\bar{Y}^i$, as the answer with the lowest energy that is different (or substantially different) from the correct answer $\bar{Y}^i = \operatorname{argmin}_{y \in \mathcal{Y}, y \neq Y^i} E(W, y, X)$. We can now define the *hinge loss*:

$$L_{\text{hinge}}(W, Y^i, X^i) = [m + E(W, Y^i, X^i) - E(W, \bar{Y}^i, X^i)]^+$$

where m is a positive *margin* parameter, and $[v]^+ = \max(0, v)$. This loss attempts to make the energy of the desired answer lower than the energy of the most offending incorrect answer by at least m. Several models have been proposed that use loss functions of this type, including the Support Vector Markov Model [1], and the Maximum Margin Markov Networks [28]. Other margin-like losses have been used for face detection and pose estimation, as well as for manifold learning [24, 11].

2.2. Architectures for Structured Problems

Training an EBM is particularly simple in the special case where the energy is a linear function of the parameters: $E(W, Y, X) = W^T F(X, Y)$, where $F(X, Y)$ is a feature vector extracted by a suitably designed preprocessor. Minimizing any of the above losses using a gradient-based method become an extremely simple problem. Three methods that have attracted a considerable amount of attention in the ML and NLP communities in the last few years can essentially be viewed as linearly parameterized energies with different loss functions. Collins's natural language parsers use the perceptron loss [7], the Conditional Random Field model (CRF) uses the negative log likelihood loss [15], while the Support Vector Markov Model [1] and the Maximum Margin Markov Net [28] use variations of the hinge loss.

Although the linear parameterization has the apparent advantage of making the loss functions convex, the resulting models are "shallow", and hence are limited in their ability to learn complex tasks efficiently. To many observers in the speech and handwriting communities, limiting the discussion to linearly parameterized models seems like somewhat of a throwback to the past. Discriminative handwriting recognition systems that are trained at the word level, that use highly complex non-linearly parameterized energy functions have been developed and commercially deployed (for automated check reading) since the mid 90's [18]. These systems use "deep architectures" that integrate feature extraction and classification in one trainable module (a convolutional network).

3. Training Deep Architectures

In the introduction, we alluded to the limitations of shallow architectures, such as SVMs and other kernel methods, and to the potential advantages of deep architectures. Deep architectures are composed of multiple layers of non-linear modules, each of which is dependent upon a set of trainable parameters. Conversely, shallow architectures comprise at most two layers of trainable non-linear functions. The main problem with shallow architectures is that they may require an exponential number of elements in the first layer to implement certain classes of functions. It has been argued that some of the tasks that are inefficiently implemented by shallow architectures include invariant shape classification, particularly if background clutter is present [4]. A good experimental demonstration of this is shown in table 1. The task is to recognize the category of generic objects from the NORB dataset [14]. The NORB dataset contains 50 toys from 5 categories under numerous different viewpoints, illuminations, and background clutter. The categories are human figures, animals, airplanes, cars, and trucks. 25 objects are used for training, with 9720 views each, for a total of 291,600 training samples including 9720 background images. The 25 other objects are used for testing, with 1944 views each, for a total of 58,320 test samples, including 1944 background images. An SVM with Gaussian kernel, and a convolutional net were trained on this dataset by feeding them with raw pixel images (roughly 100x100 pixels).

	SVM	Convolutional Net		
test error	43.3%	16.38%	7.5%	7.2%
training time (min*GHz)	10,944	420	2,100	5,880
sample test time (sec*GHz)	2.2	0.04		

Table 1. *Testing error rates and training/testing timings on the NORB dataset. The CPU times are normalized to hypothetical 1GHz single CPU. The convolutional nets have multiple results with different numbers of training epochs.*

After carefu adjustment of the SVM parameters to obtain the best result (kernel width = 10^4, and $C = 40$), 5% of the training samples were support vectors. The results clearly show that the SVM is overwhelmed by the complexity of the task. This is because *an SVM is little more than a glorified template matcher*. Its first layer (the kernels) merely compares the incoming vector to the support vectors from the training set. The matching scores are then linearly combined. By contrast, the 6-layer convolutional net produce low error rates, and learns the task relatively quickly. This occurs despite the fact that the SVM loss is convex while the convolutional net's loss is not.

One problem however, is the ravenous apetite of supervised learning for labeled training samples. The Deep Belief Network approach allows us to reduce the need for labeled samples by sequentially training all the layers but the last one in unsupervised mode. In his DBN work, Hinton uses a so-called Restricted Boltzmann Machine (RBM) for each layer. In an RBM, the hidden layer is composed of binary stochastic units, from which the input is reconstructed. The loss function is the negative log likelihood of the input data. Unfortunately, the second term of the loss is intractable, and one must resort to a Markov chain Monte Carlo method to approximate it by sampling. The method is dubbed "Contrastive Divergence". Such a deep network, after fine-tuning with back-propagation reaches less than 1% error on MNIST, which is the record among knowledge-free methods.

In the last year, our group has proposed an alternative method for learning such a deep hierarchy of features in an unsupervised manner [23, 25, 26]. Our method constrains the feature vectors at each layer to be *sparse and overcomplete*, because of the well-known advantages to such representations for recognition. The method was used to pretrain the convolutional filters of a convolutional net applied to the MNIST dataset of handwritten numerals. The feed-forward part of the network was then refined with backpropagation, yielding a record 0.39% error [23]. In a more recent work, a method was proposed to learn a deep hierarchy of *increasingly invariant features*, by including the "feature pooling" layer of a convolutional net into the unsupervised module [25]. Trained with this method, an extremely compact convolutional net achieved 54% recognition rate on the Caltech-101 object recognition dataset with 101 categories. For comparison, the same architecture trained entirely supervised yields a disappointing 22% recognition rate with 30 training samples per class.

4. Conclusion

The main advantages of the EBM approach are that: 1. the absence of normalization gives us more flexibility in the choice of energy functions. We are not limited to functions that can be normalized; 2. we are free to use other loss functions than the one prescribed by the probabilistic approach (the negative log likelihood loss and its variations). In particular, we are free to use loss functions that do not involve the evaluation of an intractable integral (or sum).

The EBM approach provides a simple conceptual framework with which one can describe many of the discriminative models recently developed in the ML and NLP communities, and not-so-recently developed in the handwriting and speech recognition communities.

One long term goal of Machine Learning (at least for some of us in the ML community), is to devise sufficiently general and powerful learning methods that can learn an entire recognition task from end to end with a minimal amount of labeled samples. We hinted at the fact that currently popular "shallow" models, such as kernel methods, fall short. If we want to scale the applicability of learning methods to AI-like tasks, we must concentrate our effort on solving the still-unsolved "deep learning problem".

References

[1] Y. Altun, M. Johnson, and T. Hofmann. Loss functions and optimization methods for discriminative learning of label sequences. In *Proc. EMNLP*, 2003.

[2] L. Bahl, P. Brown, P. de Souza, and R. Mercer. Maximum mutual information estimation of hidden markov model parameters for speech recognition. In *Proceedings of Acoustics, Speech, and Signal Processing Conference*, pages 49–52, 1986.

[3] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *NIPS*. MIT Press, 2007.

[4] Y. Bengio and Y. LeCun. Scaling learning algorithms towars ai. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large-Scale Kernel Machines*. MIT Press, 2007.

[5] L. Bottou. *Une Approche théorique de l'Apprentissage Connexionniste: Applications à la Reconnaissance de la Parole*. PhD thesis, Université de Paris XI, 91405 Orsay cedex, France, 1991.

[6] T. Breuel. A system for the off-line recognition of handwritten text. In *Proceedings of 12th International Conference on Pattern Recognition*, pages 129–34 vol.2, 1994.

[7] M. Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proc. EMNLP*, 2002.

[8] J. S. Denker and C. J. Burges. Image segmentation and recognition. In *The Mathematics of Induction*. Addison Wesley, 1995.

[9] X. Driancourt, L. Bottou, and G. P. Comparison and cooperation of several classifiers. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, 1991.

[10] C. Garcia and M. Delakis. A neural architecture for fast and robust face detection. *IEEE-IAPR Int. Conference on Pattern Recognition*, pages 40–43, 2002.

[11] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *Proc. Computer Vision and Pattern Recognition Conference (CVPR'06)*. IEEE Press, 2006.

[12] G. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[13] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504 – 507, July 2006.

[14] F.-J. Huang and Y. LeCun. Large-scale learning with svm and convolutional nets for generic object categorization. In *Proc. Computer Vision and Pattern Recognition Conference (CVPR'06)*. IEEE Press, 2006.

[15] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. International Conference on Machine Learning (ICML)*, 2001.

[16] Y. LeCun and Y. Bengio. word-level training of a handwritten word recognizer based on convolutional neural networks. In IAPR, editor, *Proc. of the International Conference on Pattern Recognition*, volume II, pages 88–92, Jerusalem, October 1994. IEEE.

[17] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, Winter 1989.

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[19] Y. LeCun, S. Chopra, R. Hadsell, R. Marc'Aurelio, and F. Huang. A tutorial on energy-based learning. In G. Bakir et al., editor, *Predicting Structured Data*.

[20] Y. LeCun, F.-J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of CVPR'04*. IEEE Press, 2004.

[21] Y. LeCun, U. Muller, J. Ben, E. Cosatto, and B. Flepp. Off-road obstacle avoidance through end-to-end learning. In *Advances in Neural Information Processing Systems (NIPS 2005)*. MIT Press, 2005.

[22] A. Ljolje, Y. Ephraim, and L. R. Rabiner. Estimation of hidden markov model parameters by minimizing empirical error rate. In *Proc. of International Conference on Acoustics, Speech, and Signal Processing*, pages 709–712, April 1990.

[23] R. Marc'Aurelio, C. Poultney, S. Chopra, and Y. LeCun. Efficient learning of sparse representations with an energy-based model. In J. P. et al., editor, *Advances in Neural Information Processing Systems (NIPS 2006)*. MIT Press, 2006.

[24] M. Osadchy, Y. LeCun, and M. Miller. Synergistic face detection and pose estimation with energy-based models. *Journal of Machine Learning Research*, 8:1197–1215, May 2007.

[25] M. Ranzato, F. Huang, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proc. Computer Vision and Pattern Recognition Conference (CVPR'07)*. IEEE Press, 2007.

[26] M. Ranzato and Y. LeCun. A sparse and locally shift invariant feature extractor applied to document images. In *Proc. ICDAR*, 2007.

[27] P. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document anal ysis. In *Proceedings of ICDAR 2003*, pages 958–962, 2003.

[28] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Proc. NIPS*, 2003.