TTIC 31230, Fundamentals of Deep Learning

David McAllester, Autumn 2020

Deep Graphical Models

aka, Energy Based Models

Energy Based Models

Energy based models are an alternative to autoregressive models.

An energy based model computes a score and the distributions is defined by an exponentially large softmax.

$$P_s(\hat{\mathcal{Y}}) = \underset{\hat{\mathcal{Y}}}{\text{softmax}}\ s(\hat{\mathcal{Y}}) \quad \text{all possible } \hat{\mathcal{Y}}$$

$$\text{cross-entropy loss} = -\ln P_s(\mathcal{Y}) \quad \text{gold label (training label) } \mathcal{Y}$$

Of course we cannot directly compute the exponentially large softmax distribution.

Graphical Models

Graphical models are a form of energy based model where the energy is computed by summing up local energies.

A subclass of graphical models allow the probability $P_s(\mathcal{Y})$ to be computed efficiently by dynamic programming.

Parsing and CKY

Consider the case where x is a sentence and y is a parse tree for x. There are exponentially many possible parse trees for a given sentence.

The Cocke–Kasami–Younger (CKY) algorithm is a dynamic programming algorithm for finding the most probably parse under a certain family of energy based models.

CKY is still the most accurate way to parse sentences where the energy based model is computed by a deep network.

Speech Recognition and CTC

Consider the case where x is the sound wave from a microphone and y is the transcription into written language. There are clearly exponentially many possible transcriptions.

Connectionist Temporal classification (CTC) is a dynamic programming algorithm for finding the most likely output under a certain family of energy based models.

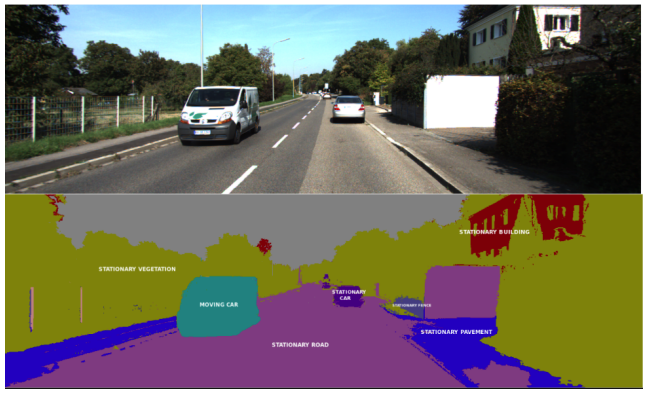
CTC is still used in many speech recognition systems.

Semantic Segmentation

We want to assign each pixel to one of L semantic classes.

For example "person", "car", "building", "sky" or "other".

Semantic Segmentation



Although semantic segmentation is not currently done with energy based models, perhaps it should be.

Semantic segmentation is simpler than parsing or speech recognition and will be used as a simple example of energy based models.

The graphical models historically used for semantic segmentation do not support dynamic programming solutions.

Notation

x is an input (e.g. an image).

$\hat{\mathcal{Y}}[N]$ is a potential structured label for x — a vector $\hat{\mathcal{Y}}[0], \ldots, \hat{\mathcal{Y}}[N-1]$ with $\hat{\mathcal{Y}}[n]$ an integer in $\{1, \ldots, L\}$.

For example n might range over the pixels of an image and $\hat{\mathcal{Y}}[n]$ names one of L a semantic labels for pixel n.

$\mathcal{Y}[N]$ is the gold label for input x — the structured label assigned to x in the training data.

Compactly Representing Scores on Exponentially Many Labels

We will call n a "node".

We will compute a "node potential" tensor $s^N[N, L]$ that holds NL scores — a score for each possible assignment of a label ℓ to n.

We assume a set E of "edges" with $E \subseteq N \times N$.

We will compute an "edge potential" tensor $s^E[E, L, L]$ that holds ELL scores — a score for each assignment of class labels ℓ_1 and ℓ_2 to the two nodes in the edge e.

Computing Scores

$$s(\hat{\mathcal{Y}}) = \sum_{n} s^N[n, \hat{\mathcal{Y}}[n]] + \sum_{\langle n,m\rangle \in E} s^E[\langle n, m\rangle, \hat{\mathcal{Y}}[n],\ \hat{\mathcal{Y}}[m]]$$

The exponential softmax for scores of this form are intractible in general — the partition function Z_s requires summing over an exponentially large set and computing $P_s(\mathcal{Y})$ is $\#P$ hard.

Computing the Node and Edge Potential Tensors

For input x we use a network to compute the score tensors.

$$s^N[N, L] = f_\Phi^N(x)$$

$$s^E[E, L, L] = f_\Phi^E(x)$$

Back-Propagation Through Exponential Softmax

$$
\begin{aligned}
s^N[I, L] &= f^N_\Phi(x) \\
s^E[E, L, L] &= f^E_\Phi(x)
\end{aligned}
$$

$$
s(\hat{\mathcal{Y}}) = \sum_n s^N[n, \hat{\mathcal{Y}}[n]] + \sum_{\langle n, m \rangle \in \text{Edges}} s^E[\langle n, m \rangle, \hat{\mathcal{Y}}[n],\ \hat{\mathcal{Y}}[m]]
$$

$$
P_s(\hat{\mathcal{Y}}) = \underset{\hat{\mathcal{Y}}}{\text{softmax}}\ s(\hat{\mathcal{Y}}) \quad \text{all possible } \hat{\mathcal{Y}}
$$

$$
\mathcal{L} = -\ln P_s(\mathcal{Y}) \quad \text{gold label } \mathcal{Y}
$$

We want the gradients $s^N.\text{grad}[N, L]$ and $s^E.\text{grad}[E, L, L]$.

END