TTIC 31230, Fundamentals of Deep Learning

David McAllester, Autumn 2020

Deep Graphical Models

aka, Energy Based Models

Distributions on Exponentially Large Sets

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{(x,y)\sim \mathrm{Pop}} \; -\ln \; P(y|x)$$

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{y\sim \mathrm{Pop}} \; -\ln \; P(y)$$

The structured case: $y \in \mathcal{Y}$ where $\mathcal{Y}$ is discrete but iteration over $\hat{y} \in \mathcal{Y}$ is infeasible.

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.

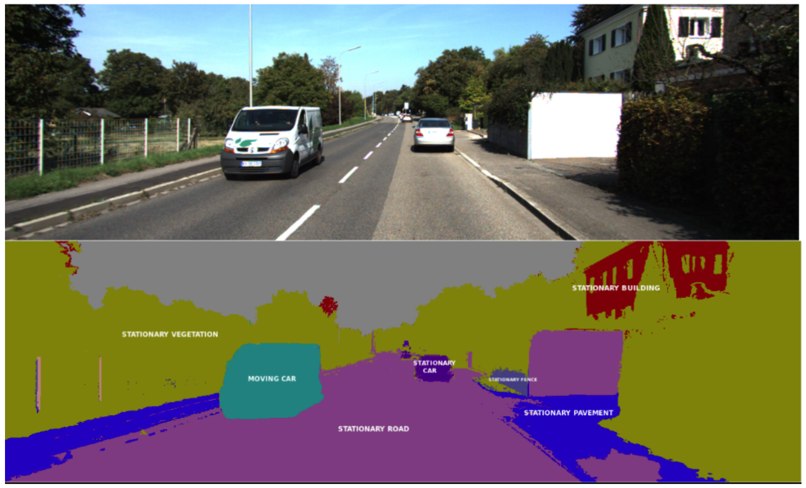
Speech Recognition and CTC

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 the most accurate way to do speech recognition where the energy based model is computed by a deep network.

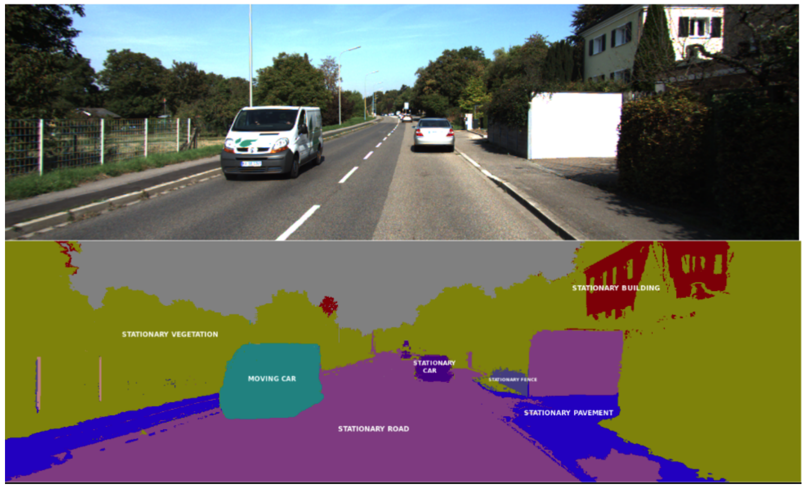
Semantic Segmentation



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

For example “person”, “car”, “building”, “sky” or “other”.

Semantic Segmentation



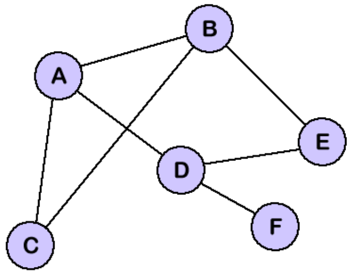
Although semantic segmentation is not currently done with energy based models, perhaps it should be.

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

Constructing a Graph

We construct a graph whose nodes are the pixels and where there is an edges between each pixel and its four nearest neighboring pixels.

Labeling the Nodes of a Graph



$\hat{y}$ assigns a semantic class $\hat{y}[n]$ to each node (pixel) n.

We assign a score to $\hat{y}$ by assigning a score to each node and each edge of the graph.

$$s(\hat{y}) = \underbrace{\sum_{n \in \mathrm{Nodes}} s^N[n, \hat{y}[n]]}_{\text{Node Scores}} + \underbrace{\sum_{\langle n,m\rangle \in \mathrm{Edges}} s^E[\langle n, m\rangle, \hat{y}[n], \hat{y}[m]]}_{\text{Edge Scores}}$$

Using Deep Networks

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

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

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

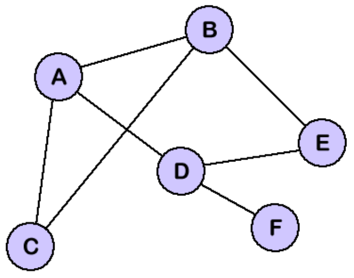
Exponential Softmax

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

$$\text{for } \hat{y} \;\; P_s(\hat{y}) \;=\; \text{softmax}_{\hat{y}} \;\; s(\hat{y}) \quad \text{all possible } \hat{y}$$

$$\mathcal{L} \;=\; -\ln P_s(y) \qquad \text{gold label (training label) } y$$

Exponential Softmax is Typically Intractable



$\hat{y}$ assigns a label $\hat{y}[n]$ to each node n.

$s(\hat{y})$ is defined by a sum over node and edge tensor scores.

$P_s(\hat{y})$ is defined by an exponential softmax over $s(\hat{y})$.

Computing Z in general is #P hard (there is an easy direct reduction from SAT).

Compactly Representing Scores on Exponentially Many Labels

The tensor $s^N[N, Y]$ holds NY scores.

The tensor $s^E[E, Y, Y]$ holds EY^2 scores where e ranges over edges $\langle n, m \rangle \in \text{Edges}$.

Back-Propagation Through Exponential Softmax

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

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

$$P_s(\hat{y}) = \operatorname*{softmax}_{\hat{y}}\ s(\hat{y}) \quad \text{all possible } \hat{y}$$

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

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

END