# CPSC 440/540: Advanced Machine Learning

## End-to-End Learning, Exponential Families

Danica Sutherland (building on materials from Mark Schmidt)

University of British Columbia

Winter 2023

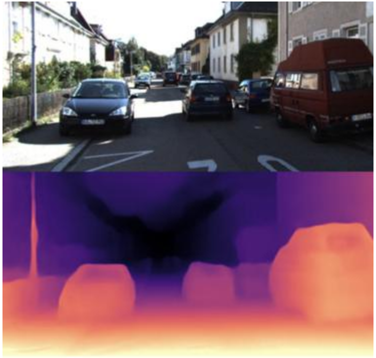
# Last time: Rejection+Importance Sampling, Laplace Approximation

- Mostly, we want to estimate $\mathbb{E}_{X\sim p} f(X)$ for some $f$
  - Indicators of events, conditional probabilities, mean / variance, ...
- Rejection sampling finds exact samples from $p$, then $\mathbb{E}_{X\sim p} f(x) \approx \sum_i \frac{1}{n} f(x^i)$
  - Propose from $q(x)$, know $M \ge \max_x \frac{\tilde{p}(x)}{q(x)}$; then accept with probability $\frac{\tilde{p}(x)}{M q(x)}$
  - High rejection rate if $q$ "far from" $p$ (e.g. in high dimensions)
- Importance sampling gets weighted "samples", then $\mathbb{E}_{X\sim p} f(x) \approx \sum_i w^i f(x^i)$
  - Sample $x^i \overset{iid}{\sim} q(x)$, weight $w^i = p(x^i)/(nq(x^i))$
  - If we only know $\tilde{w}^i = \tilde{p}(x^i)/q(x^i)$, self-normalized IS uses $\hat{w}^i = \tilde{w}^i/(\sum_j \tilde{w}^j)$
  - High variance (and, for self-norm, high bias) if $q$ far from $p$ (e.g. in high dimensions)
- Laplace approximation with a Gaussian $q$, then $\mathbb{E}_{X\sim p} f(X) \approx \mathbb{E}_{X\sim q} f(X)$
  - Find $x^* = \arg\max_x p(x)$, use $q = \mathcal{N}\left(x^*, \left(\nabla_x^2[-\log p(x)]\big|_{x^*}\right)^{-1}\right)$
  - Fast but can be very bad if $p$ doesn't look like a Gaussian near its mode

# Outline

# Motivating Problem: Depth Estimation from Images

- We want to predict “distance to car” for each pixel in an image.



https://paperswithcode.com/task/3d-depth-estimation

- We might consider using fully-convolutional networks.
  - But we now have multiple continuous labels.

# Neural Network with Continuos Outputs

- Standard neural network with multiple continuous outputs (3 hidden layers):

$$\hat{y}^i = Vh(W^3h(W^2h(W^1x^i))), \quad \text{so} \quad \hat{y}_c^i = v_c^T h(W^3h(W^2h(W^1x^i))).$$

- Standard training objective is to minimize squared error,

$$f(W^1, W^2, W^3, V) = \frac{1}{2}\sum_{j=1}^{n}\sum_{c=1}^{k}(y_c^i - \hat{y}_c^i)^2.$$

- This corresponds to MLE in a network that outputs the mean of a Gaussian,

$$y^i \sim \mathcal{N}(\hat{y}^i, \mathbf{I}).$$

- As usual, we only need to change the last layer to change output type.

# Neural Networks with Covariances

bonus!

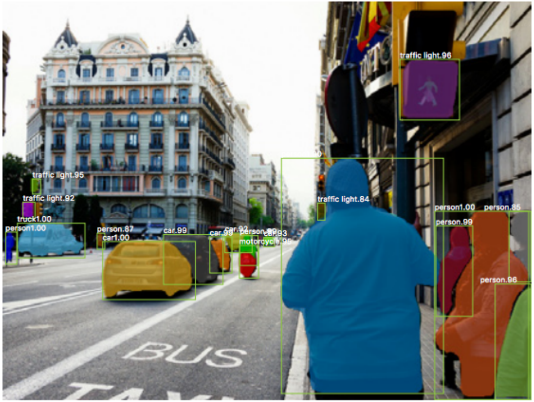
- The neural network could also parameterize the variance,

$$y^i \sim \mathcal{N}(\hat{y}^i, S(W^3 h(W^2 h(W^1 x^i)))),$$

  where the function $S$ transforms the hidden layer into a positive-definite matrix.
  - So inferences over multiple variables will capture the label's pairwise correlations.
    - For depth estimation, neighbouring pixels are likely to have similar depths.
- Common choices for $S$:
  - $S$ parameterizes a diagonal matrix $D$ (may output $\log(\sigma_c)$ values to make positive).
  - $S$ parameterizes a square root matrix $A$, such that $\Sigma = AA^T$.
- We could also consider Bayesian neural networks.
  - Where you might use a Laplace approximation of the posterior.
    - Though the matrix $\nabla^2 f(W^3, W^2, W^1, V)$ may be too large and will be singular.

# Object Localization

- Object localization is task of finding locations of objects:
  - Input is an image.
  - Output is a bounding box for each object (among predefined classes).

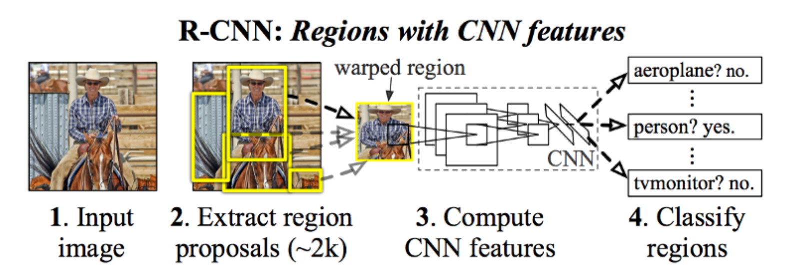



https://blog.athelas.com/a-brief-history-of-cnns-in-image-segmentation-from-r-cnn-to-mask-r-cnn-34ea83205de4

# Region Convolutional Neural Networks: "Pipeline" Approach

bonus!

- Early approach (region CNN) resemble classic computer vision "pipelines":
  1. Propose a bunch of potential boxes (based on segmenting image in various ways).
  2. Compute features of each box using a CNN (after re-shaping box to standard size).
  3. Classify boxes using SVMs (max pool among regions with high overlap).
  4. Refine each box using linear regression on CNN features.
     - 4 continuous outputs: center x-coordinate, center y-coordinate, log-width, log-height.



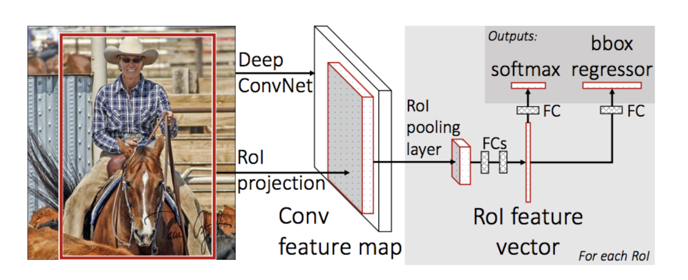

https://arxiv.org/pdf/1311.2524.pdf

- Improved on state of the art, but slow and there are 4 parts to train.

# Fast R-CNNs

bonus!

- R-CNN was quickly replaced by fast R-CNN:
  - Propose a bunch of potential bounding boxes (same as before).
  - Apply CNN to whole image, then get features of bounding boxes.
    - Faster than applying CNN to 2000 candidate regions.
  - Make softmax (over $k + 1$ classes) and bounding box regression part of network.
    - More accurate since are parts are trained together.



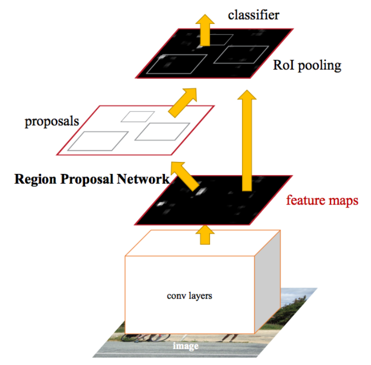

https://arxiv.org/pdf/1504.08083.pdf

- Most parts trained together, but bounding box proposals do not use encoding.

# Faster R-CNNs

bonus!

- Faster R-CNNs made generating bounding boxes part of the network.
  - Uses region-proposal network as part of network to predict potential bounding boxes.
  - Many implementation details required to get it working.




https://arxiv.org/pdf/1506.01497.pdf

- With all steps being part of one network, this called an end-to-end model.

# YOLO: You Only Look Once

bonus!

- A more-recent variant that further speeds things up is YOLO:

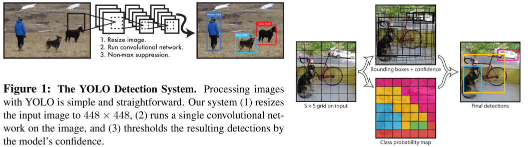



**Figure 1: The YOLO Detection System.** Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448 × 448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.

**Figure 2: The Model.** Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts $B$ bounding boxes, confidence for those boxes, and $C$ class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

https://arxiv.org/pdf/1506.02640.pdf

  - Divides image into grid.
  - Directly predict properties for a fixed number of bounding boxes for grid box:
    - Probability that box is an object (for pruning set of possible boxes).
    - Box x-coordinate, y-coordinate, width, height.
    - Class of box (no separate phase of "proposing boxes" and "classifying boxes").
  - Max pooling ("non-max suppression").
- Reasonably-accurate real-time object detection (with fancy-enough hardware).

# Instance Segmentation and Pose Estimation

bonus!

- Can add extra predictions to these networks.
- For example, mask R-CNNs add instance segmentation and/or pose estimation:

https://arxiv.org/pdf/1703.06870.pdf

- Instance segmentation applies binary mask to bounding boxes (pixel labels).
- Pose estimation predicts continuous joint keypoint locations.

# End-to-End Computer Vision Models

- Key ideas behind end-to-end systems:
  1. Write each step as a differentiable operator.
  2. Train all steps using backpropagation and stochastic gradient.

- Has been called differentiable programming.

- There now exist end-to-end models for all the standard vision tasks.
  - Depth estimation, pose estimation, optical flow, tracking, 3D geometry, and so on.
  - A bit hard to track the progress at the moment.
  - A survey of $\approx 200$ papers from 2016 (has only grown since):
    - `http://www.themtank.org/a-year-in-computer-vision`
- Pose estimation video: `https://www.youtube.com/watch?v=pW6nZXeWlGM`
- Making 60-fps high-resolution colour version of videos from 120 year ago:
  - `https://www.youtube.com/watch?v=YZuP41ALx_Q`

# End of Part 3 ("Gaussian Variables"): Key Concepts

- We discussed continuous density estimation with multivariate Gaussians.
  - Parameterized by mean vector and positive definite covariance matrix.
  - Assumes distribution is uni-modal, no outliers, untruncated.
    - And symmetric around principle axes.
  - "Gaussianity" is preserved under many operations.
    - Addition, marginalization, conditionining, product of densities.

- We discussed conditional independence in Gaussians.
  - Models correlations between variables where $\Sigma_{ij} \neq 0$.
    - Diagonal covariance corresponds to assuming variables all variables are independent.
  - We define a graph based on the $\Theta_{ij}$ values.
    - If variables are blocked in graph, implies conditional independence.

# End of Part 3 ("Gaussian Variables"): Key Concepts

- We discussed several methods for sampling and/or Monte Carlo:
  - Inverse transform method uses inverse of CDF to sample continuos densities.
  - Rejection sampling rejects samples from a simpler distribution.
  - Importance sampling reweights samples from a simpler distribution.

- We discussed learning in Gaussians.
  - Closed-form MLE given by data's mean and variance.
  - Conjugate prior for mean in Gaussian.
  - Adding a scaled identity matrix to MLE gives positive-definite estimate.
  - Graphical Lasso allows learning sparse conditional independence graph.

- Gaussian discriminant analysis is generative classifer with Gaussian classes.
  - Does not need naive Bayes assumption.

# End of Part 3 ("Gaussian Variables"): Key Concepts

- We discussed regression.
  - Supervised learning with continuous outputs.
  - Least squares with L2-regularization assumes Gaussian likelihood and prior.

- We discussed Bayesian linear regression.
  - Gives confidence in predictions.
  - Empirical Bayes can be used to set many hyper-parameters.
    - Automatic relevance determination: prefers simpler models that fit data well.
  - Laplace approximation can be used in non-conjugate settings.
    - Special case of a variational inference method (approximate with simpler distribution).

- We discussed end-to-end learning.
  - Try to write each step as a differentiable operation.
  - Train entire network with backprop and SGD.
    - We illustrated this with evolution of object localization in vision.

# Outline

# Previously: Density Estimation with Categorical/Gaussian Distributions

- We have discussed density estimation with categorical and Gaussian distribution.
  - Binary is special case of categorical.
- These distributions have a lot of nice properties for learning/inference.
  - NLL is convex, and MLE has closed-form (statistics in training data).
  - A conjugate prior exists, so posterior is prior with "updated hyper-parameters."
- But these distributions make restrictive assumptions:
  - Categorical assumes categories are unordered, non-hierarchical, and finite.
  - Gaussian assumes symmetry, full support, no outliers, uni-modal.
- Many alternatives to categorical/Gaussian exist (examples later).
  - Alternatives that are in the exponential family maintain nice properties.

# Exponential Family: Definition

- General form of exponential family likelihood for data $x$ with parameters $\theta$ is

$$p(x \mid \theta) = \frac{h(x)\exp(\eta(\theta)^\mathsf{T} s(x))}{Z(\theta)}.$$

- The value $s(x)$ is the vector of sufficient statistics.
  - $s(x)$ tells us everything that is relevant to $\theta$ about data $x$.
- The parameter function $\eta$ controls how parameters $\theta$ interact with the statistics.
  - We'll focus a lot on $\eta(\theta) = \theta$, which is called the canonical form.
- The support function $h$ contains terms that don't depend on $\theta$.
  - Also called the base measure.
- The normalizing constant $Z$ ensures it sums/integrates to 1 over $x$.
  - Also called the partition function.

# Bernoulli as Exponential Family

- Is Bernoulli in the exponential family for some parameters $w$?

$$p(x \mid \theta) = \theta^x (1-\theta)^{1-x}\ \mathbb{1}(x \in \{0,1\}) \stackrel{?}{=} \frac{h(x)\exp(\eta(\theta)^T F(x))}{Z(\theta)}.$$

- To get an exponential, take log of exp (cancelling operations),

$$\begin{aligned} p(x \mid \theta) &= \theta^x (1-\theta)^{1-x}\ \mathbb{1}(x \in \{0,1\}) = \exp(\log(\theta^x (1-\theta)^{1-x}))\ \mathbb{1}(x \in \{0,1\}) \\ &= \exp(x \log\theta + (1-x)\log(1-\theta))\ \mathbb{1}(x \in \{0,1\}) \\ &= (1-\theta)\exp\left(x \log\left(\frac{\theta}{1-\theta}\right)\right)\ \mathbb{1}(x \in \{0,1\}). \end{aligned}$$

- The sufficient statistic is $s(x) = x$ and normalizing constant is $Z(\theta) = 1/(1-\theta)$.
- The parameter is $\eta(\theta) = \log(\theta/(1-\theta))$ (the log odds).
  - Not in canonical form. Canonical form would use log odds directly as the parameter.
- The support function is $h(x) = \mathbb{1}(x \in \{0,1\})$ – says if we're "in the support".
- There are also other ways to write Bernoulli as an exponential family.

# Gaussian as Exponential Family

- Writing univariate Gaussian as an exponential family:

$$
\begin{aligned}
p(x \mid \mu, \sigma^2) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-(x-\mu)^2/2\sigma^2\right) \\
&= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-x^2/2\sigma^2 + \mu x/\sigma^2 - \mu^2/2\sigma^2\right) \\
&= \frac{1}{\sqrt{2\pi}} \frac{\exp\left(-\mu^2/2\sigma^2\right)}{\sigma} \exp\left(\begin{bmatrix} \mu/\sigma^2 \\ -1/2\sigma^2 \end{bmatrix}^T \begin{bmatrix} x \\ x^2 \end{bmatrix}\right).
\end{aligned}
$$

- The sufficient statistics are $x$ and $x^2$, and canonical params are $\mu/\sigma^2$ and $-1/2\sigma^2$
- The normalizing constant is $\sigma \exp(\mu^2/2\sigma^2)$, and support is $1/\sqrt{2\pi}$.

- Again, there is more than one way to represent as an exponential family.
  - If $\sigma^2$ is considered fixed, then $x/\sigma^2$ is the sufficient statistic and $\mu$ is canonical.

# Learning with Exponential Families

- With $n$ IID examples and canonical parameters $\theta$, the likelihood is

$$\begin{aligned} p(\mathbf{X} \mid \theta) &= \prod_{i=1}^{n} h(x^i) \frac{\exp(\theta^{\mathsf{T}} s(x^i))}{Z(\theta)} \\ &= \frac{1}{Z(\theta)^n} \exp\left(\theta^{\mathsf{T}} \sum_{i=1}^{n} s(x^i)\right) \prod_{j=1}^{n} h(x^i) \\ &= \frac{\exp(\theta^{\mathsf{T}} s(\mathbf{X}))}{Z(\theta)^n} \prod_{j=1}^{n} h(x^i), \end{aligned}$$

  where the sufficient statistics are $s(\mathbf{X}) = \sum_{i=1}^{n} s(x^i)$.

- $s(\mathbf{X})$ contain everything relevant for learning – can throw away the actual data.
  - For Gaussians, only knowledge of data we need is $\sum_{i=1}^{n} x^i$ and $\sum_{i=1}^{n} (x^i)^2$.
  - No point in using SGD: you just compute $s$ on each example once.
  - Exponential families are the *only* class of distributions with a finite sufficient statistic.

# Learning with Exponential Families

- With IID data and canonical $\theta$, NLL is $f(\theta) = -\theta^\mathsf{T} s(\mathbf{X}) + n \log Z(\theta) + \text{const}$.
- The gradient divided by $n$ (average NLL) for a feature $j$ is

$$
\begin{aligned}
\frac{1}{n}\nabla_{\theta_j} f(\theta) &= -\frac{1}{n} s_j(\mathbf{X}) + \frac{1}{Z(\theta)} \nabla_{\theta_j} Z(\theta) \\
&= -\frac{1}{n} s_j(\mathbf{X}) + \frac{1}{Z(\theta)} \nabla_{\theta_j} \int h(x) \exp\left(\theta^\mathsf{T} s(x)\right) \mathrm{d}x \quad \text{(use } \textstyle\sum \text{ for discrete } x\text{)} \\
&= -\frac{1}{n} s_j(\mathbf{X}) + \int_x h(x) \frac{\exp(\theta^\mathsf{T} s(\mathbf{X}))}{Z(\theta)} s_j(\mathbf{X}) \, \mathrm{d}x \qquad \text{(w/ conditions)} \\
&= -\frac{1}{n} s_j(\mathbf{X}) + \int_x p(x \mid \theta) s_j(x) \mathrm{d}x \\
&= -\mathbb{E}_{X \sim \text{data}}[s_j(X)] + \mathbb{E}_{X \sim \text{model}}[s_j(X)].
\end{aligned}
$$

- The stationary points where $\nabla f(\theta) = 0$ correspond to moment matching:
  - Set parameters $\theta$ so that expected sufficient statistics equal to statistics in data.
  - This is the source of the simple/intuitive closed-form MLEs we've seen so far.

# Convexity and Entropy in Exponential Families

bonus!

- If you take the second derivative of the NLL you get

$$\nabla^2 f(\theta) = \mathrm{Cov}[s(X)],$$

  the covariance of the sufficient statistics.
  - Covariances are positive semi-definite, $\mathrm{Cov}[s(X)] \succeq 0$, so NLL is convex.
  - This is why "setting the gradient to zero and solve for $\theta$" gives MLE.
- Higher-order derivatives give higher-order moments.
  - We call $\log(Z)$ the cumulant function.
- Can show MLE maximizes entropy over all distributions that match moments.
  - Entropy is a measure of "how random" a distribution is.
  - So Gaussian is "most random" distribution that fits means and covariance of data.
    - Or you can think of this as Gaussian makes "least assumptions".
  - Details for special case of $h(x) = 1$ in bonus slides.

# Conjugate Priors in Exponential Family

- Exponential families in canonical form are guaranteed to have conjugate priors.
- For example, we could choose a prior like

$$p(\theta \mid \alpha) \propto \frac{\exp(\theta^{\mathsf{T}}\alpha)}{Z(\theta)^k}.$$

  - $\alpha$ is "pseudo-counts" for the sufficient statistics.
  - $k$ modifies the stength of the prior ($Z$ above is normalizer for the likelihood).
  - For fixed $k$, itself an exp. family in $\theta$: $s(\theta) = \theta$, parameter $\alpha$, base measure $Z(\theta)^{-k}$.
- Then the posterior has the same form,

$$p(\theta \mid \mathbf{X}, \alpha) \propto \frac{\exp(\theta^{\mathsf{T}}(s(\mathbf{X}) + \alpha))}{Z(\theta)^{n+k}}.$$

- Prior's normalizing constant (some $\zeta_k(\alpha)$, not $Z(\theta)$) useful for Bayesian inference.
  - e.g. can derive, like before, that $p(\mathbf{X} \mid \alpha) = \zeta_k(s(x) + \alpha)/\zeta_k(\alpha) \cdot \prod_{i=1}^n h(x^i)$.

# Discriminative Models and the Exponential Family

- Going from an exponential family to a discriminative supervised learning:
  - Set canonical parameter to $w^\mathsf{T} x^i$.
  - Gives a convex NLL, where MLE tries to match data/model's conditional statistics.
  - Called generalized linear model (GLM) – see Stat 538A, Generalized Linear Models :)

- For example, consider Gaussian with fixed variance for $y^i$.
  - Canonical parameter is $\mu$, and we know setting $\mu = w^\mathsf{T} x^i$ gives least squares.

- If we start with Bernoulli for $y^i$, we obtain logistic regression.
  - Canonical parmaeter is log-odds.
  - Set $w^\mathsf{T} x^i = \log(y^i/(1 - y^i))$ and solve for $y^i$ to get sigmoid function.
    - Finally, we know "why use the sigmoid function?"

- You can obtain regression models for other settings using this kind of approach.
  - Set canonical parameters to $v^\mathsf{T} h(W^2 h(W^1 x^i))$ for neural networks.
  - Use a different exponential family to handle a different type of data.

# Examples of Exponential Families

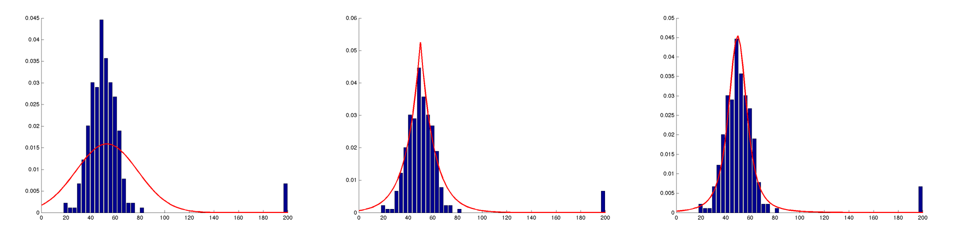
bonus!

- Bernoulli: distribution on $\{0, 1\}$.
- Categorical: distribution on $\{1, 2, \ldots, k\}$.
- Gaussian: distribution on $\mathbb{R}^d$.
- Beta: distribution on $[0, 1]$ (including uniform).
- Dirichlet: distribution on discrete probabilities.
- Wishart: distribution on positive-definite matrices.
- Poisson: distribution on non-negative integers.
- Gamma: distribution on positive real numbers.
- Many many others:
  - `en.wikipedia.org/wiki/Exponential_family#Table_of_distributions`
- . . . can even have infinite-dimensional statistics via kernel exponential families.

# Non-Examples of Exponential Families

bonus!

- Laplace and student $t$ distribution are not exponential families.



  - "Heavy-tailed": have larger probability that data is far from mean.
  - More robust to outliers than Gaussian.
- Ordinal logistic regression is not in exponential family.
  - Can be used for categorical variables where ordering matters.
- In these cases, we may not have nice properties:
  - MLE may not be intuitive or closed-form, NLL may not be convex.
  - May not have conjugate prior, so need Monte Carlo or variational methods.

# Summary

- Neural networks with continous output:
  - Typically trained using squared error, corresponding to Gaussian likelihood.
- End to end models: use a neural network for everything.
  - Each step in a vision "pipeline" as a differentiable operator; train with SGD.
- Exponential families:
  - Have sufficient statistics and canonical parameters.
  - Maximimum likelihood becomes moment matching; always have conjugate priors.
  - Can build discriminative models by using canonical parameter $s(x) = w^\mathsf{T} x$.
  - Many things (but not everything!) are exponential families.
- Next time: Markov chains!

# Convex Conjugate and Entropy

bonus!

- The convex conjugate of a function $A$ is given by

$$A^*(\mu) = \sup_{w \in \mathcal{W}} \{\mu^\mathsf{T} w - A(w)\}.$$

- E.g., if we consider for logistic regression

$$A(w) = \log(1 + \exp(w)),$$

we have that $A^*(\mu)$ satisfies $w = \log(\mu)/\log(1-\mu)$.

  - When $0 < \mu < 1$ we have

$$\begin{aligned} A^*(\mu) &= \mu\log(\mu) + (1-\mu)\log(1-\mu) \\ &= -H(p_\mu), \end{aligned}$$

    negative entropy of binary distribution with mean $\mu$.
  - If $\mu$ does not satisfy boundary constraint, $\sup$ is $\infty$.

# Convex Conjugate and Entropy

bonus!

- More generally, if $A(w) = \log(Z(w))$ for an exponential family then

$$A^*(\mu) = -H(p_\mu),$$

subject to boundary constraints on $\mu$ and constraint:

$$\mu = \nabla A(w) = \mathbb{E}[s(X)].$$

- Convex set satisfying these is called marginal polytope $\mathcal{M}$.
- If $A$ is convex (and LSC), $A^{**} = A$. So we have

$$A(w) = \sup_{\mu \in \mathcal{U}} \{w^\mathsf{T}\mu - A^*(\mu)\}.$$

and when $A(w) = \log(Z(w))$ we have

$$\log(Z(w)) = \sup_{\mu \in \mathcal{M}} \{w^\mathsf{T}\mu + H(p_\mu)\}.$$

- This can be used to derive variational methods, since we have written computing $\log(Z)$ as a convex optimization problem.

# Maximum Likelihood and Maximum Entropy

bonus!

- The maximum likelihood parameters $w$ in exponential family satisfy:

$$\begin{aligned}
&\min_{w \in \mathbb{R}^d} -w^\top s(D) + \log(Z(w)) \\
&= \min_{w \in \mathbb{R}^d} -w^\top s(D) + \sup_{\mu \in \mathcal{M}} \{w^\top \mu + H(p_\mu)\} && \text{(convex conjugate)} \\
&= \min_{w \in \mathbb{R}^d} \sup_{\mu \in \mathcal{M}} \{-w^\top s(D) + w^\top \mu + H(p_\mu)\} \\
&= \sup_{\mu \in \mathcal{M}} \{\min_{w \in \mathbb{R}^d} -w^\top s(D) + w^\top \mu + H(p_\mu)\} && \text{(convex/concave)}
\end{aligned}$$

which is $-\infty$ unless $s(D) = \mu$ (e.g., maximum likelihood $w$), so we have

$$\begin{aligned}
&\min_{w \in \mathbb{R}^d} -w^\top s(D) + \log(Z(w)) \\
&= \max_{\mu \in \mathcal{M}} H(p_\mu),
\end{aligned}$$

subject to $s(D) = \mu$.

- Maximum likelihood $\Rightarrow$ maximum entropy $+$ moment constraints.