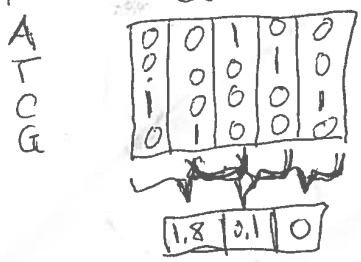
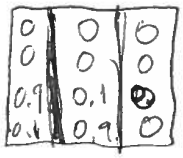
Problem Set 5: Visualizing CNN motifs as PWM

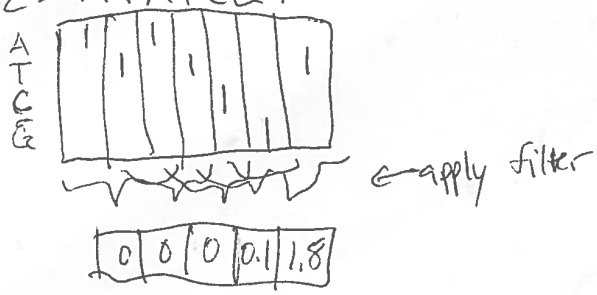
sequence1 = CGATC



filter = (3×1) (depth=1)



Sequence2 = ATATCGT



max_activation over {seq1, seq2} = 1.8

any activation $\geq \frac{1}{2}$ max_activation is "activated"

So "activated" regions from sequence1 = CGA

"activated" regions from sequence2 = CGT

Next, we form alignment and obtain PWM from Aligment.

To get PWM, first form frequency table:

Frequencies:

A	0	0	1
G	0	2	0
C	2	0	0
T	0	0	1

⇒

Fractions of each base at each position:

A	0	0	0.5
G	0	1	0
C	1	0	0
T	0	0	0.5

Convert these fractions to "log-odds" against a background model where each base occurs w/ probability = $\frac{1}{4}$

Data Augmentation:

- Not enough data $\{(x_i, y_i)\}_{i=1}^{N}$ ← too small available to get best NN performance

- We know invariants of problem (prior knowledge):
= changes to x_i which should not change y_i

- Idea: Append additional example
$(\tilde{x}_i, y_i)$ to dataset
where $\tilde{x}_i$ = variant of x_i which we know should recieve same label

- Train NN on larger dataset w/ additional examples

Ex: (1.) If x_i = sentences, can replace words with synonyms

(2.) If x_i = images, can randomly crop/rotate/translate

(3.) Can sometimes apply dropout to inputs x_i or add Gaussian noise.

(4.) (implicit augmentations)

Adversarial Augmentation: Suppose we know y_i should stay the same if x_i only slightly changes (smooth relationship).
Can use backprop gradients to modify x_i slightly such that network's prediction changes the most.
Include this $(\tilde{x}_i, y_i)$ in dataset.
← same label as x_i

Transfer Learning:

$D_1 = \{(x_i, y_i)\}_{i=1}^{N_1}$ ← Large = massive data from Task 1

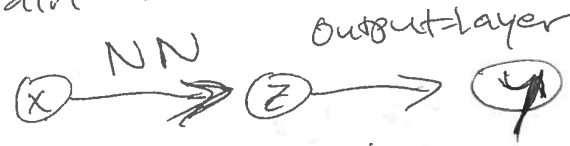
Task 2 = main task of interest

$D_2 = \{(x_j, \tilde{y}_j)\}_{j=1}^{N_2}$ ← small = small data from Task 2

x_i, x_j are same features

$y_i, \tilde{y}_j$ may be different labels!

Step 1: Train NeuralNet for Task 1:



Step 2: replace output layer with output-layer for Task 2

Step 3: Fine-tune the parameters of NN for Task 2.

Ex: For classifying biomedical images, start with NN trained on Massive object-recognition database (ImageNet)

Suppose Task 2 labels not given.

Train Task 2 NN to map $x_j \to z_j$ such that z_j and z_i (output by Task 1 NN applied to x_i) follow same statistical probability distribution.

Then apply output layer from Task 1.

Match z-distributions using divergence (eg. MMD)

Differentiable statistical divergence
1. Maximum Mean Discrepancy
2. Wasserstein Distance
3. Generative Adversarial Network

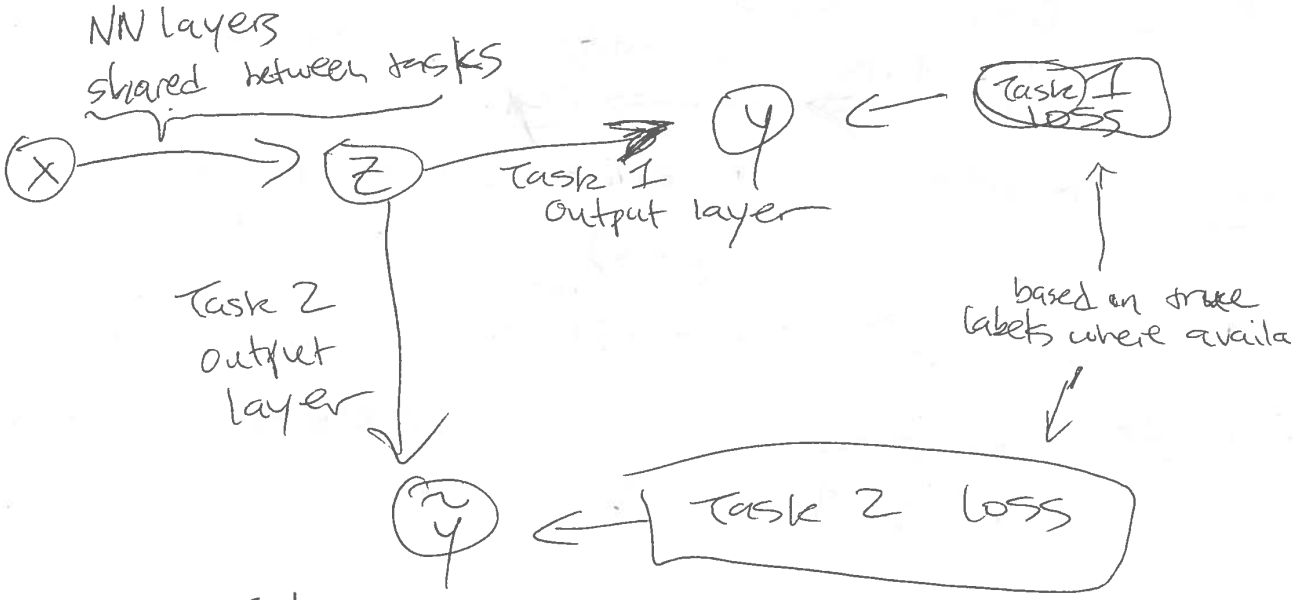
Multi-Task Learning:

2 Tasks: (x, y), $(x, \tilde{y})$

(x, y) — Task 1 ; $(x, \tilde{y})$ — Task 2

- Not enough data from either task
- We expect similarity between tasks, how can we "borrow strength" across task

Idea = Train jointly!



When only ~~Task~~ Task 1 label available, only backpropagate Task 1 loss

When only Task 2 label available, only backpropagate Task 2 loss.

If both labels availabe, backpropagate both Task 1 & 2 losses.