Final Exam: NLP @ IUST Fall 2022

Name: **Id:**

1. Multiple Choice (20 points)

For each of the following questions, circle the letter of your choice. *There is only ONE correct choice.* No explanations are required.

(a) (5 points) If your training loss increases with number of epochs, which of the following could be a possible issue with the learning process?

(A) Regularization is too low and model is overfitting

(B) Regularization is too high and model is underfitting

(C) Step size is too large

(D) Step size is too small

C. The train loss always decreases whether the model is overfitting or underfitting. If the step size is too small, the convergence is too slow, but the training loss will still go down. If the step size is too large, it may cause a bouncing effect because we skip and overshoot the optimal solution. This leads to increase in training loss and decrease in training accuracy.

(b) (5 points) The biggest advantage of neural transition-based dependency parsers over non-neural transition-based dependency parsers is that neural parsers:

(A) Choose transitions using more words in the stack and buffer

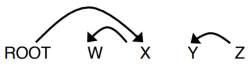
(B) Generate a larger class of dependency parses

(C) Model a grammar whereas traditional parsers do not

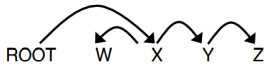
(D) Rely on dense feature representations

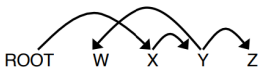
D. The main advantage of neural dependency parsers is that they offer a dense representation instead of a spare representation of the parser. Neural and traditional parsers are not different in what input information they can use, or what kinds of parses they can output (both can output any parse), but they differ in their representation of the features they use.

(c) (5 points) Which one of the following is a valid projective dependency parse?

(A) 

(B) 

(C) 

(D) 

C. In (A) Z has no head, in (B) W has two heads, and (D) is not projective because ROOT→X and Y→W cross.

(d) (5 points) Which of the following statements is INCORRECT?

(A) Recurrent neural networks can handle a sequence of arbitrary length, while feedforward neural networks can not.

(B) Training recurrent neural networks is hard because of vanishing and exploding gradient problems.

(C) Gradient clipping is an effective way of solving vanishing gradient problem.

(D) Gated recurrent units (GRUs) have fewer parameters than LSTMs.

C. Gradient clipping is only a solution for solving exploding gradient problems, not vanishing gradient ones.

2. Autoencoders (19 points)

In class, we've learned that one way to combine word vectors is to simply add them together. In this question, we'll explore another approach: using an autoencoder.

In the autoencoder, two input word vectors $x_1, x_2 \in \mathbb{R}^{D_x \times 1}$ are first concatenated into a single vector $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^{2D_x \times 1}$ and the parent vector p can be computed as:

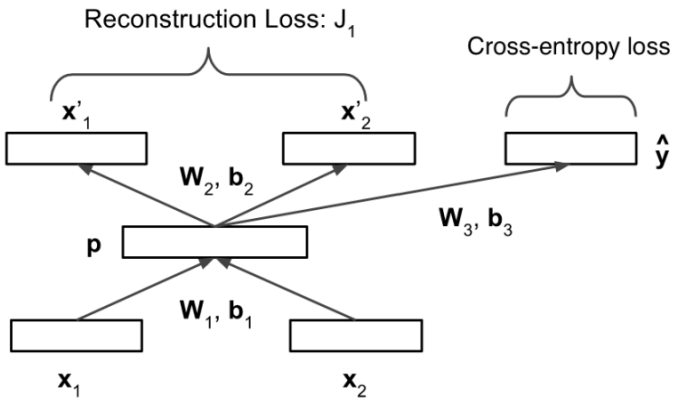


Figure 1: Illustration of an autoencoder unit to combine two vectors

$$p = ReLU(W_1 x + b) \in \mathbb{R}^{D_x \times 1} \qquad ReLU(x) = max\{x, 0\}$$

where W_1 can be decomposed as:

$$W_1 = \begin{bmatrix} W_{11} & W_{12} \end{bmatrix} \qquad W_1 x = W_{11} x_1 + W_{12} x_2$$

During training, we use the parent vector **p** to reconstruct the input vectors:

$$x'_1 = \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} = W_2 p + b_2 \in \mathbb{R}^{2D_x \times 1}$$

where $x'_1, x'_2 \in \mathbb{R}^{D_x \times 1}$ are the reconstructions. Correspondingly, a reconstruction loss that computes the Euclidean distance between inputs and reconstructions is used during training:

$$J_1 = \frac{1}{2}||x' - x|| \in \mathbb{R}$$

For sentiment classification, the parent vector is used to predict a sentiment class ˆy for the pair of the input words:

$$\hat{y} = W_3 p + b_3 \in \mathbb{R}^{D_c \times 1}$$

where $D_c = 3$ is the number of sentiment classes ('positive', 'neutral' and 'negative'). Here, the network is also trained using a cross-entropy loss:

$$J_2 = CE(y, \hat{y}) \in \mathbb{R}$$

where $y \in \mathbb{R}^{D_c \times 1}$ is the one-hot label vector with $y_k = 1$. In total, the network is trained using the loss $J = J_1 + J_2$

(a) (5 points) Give at least one example of how the sentiment predictions made by an autoencoder model would differ from one made using a bag-of-vectors model like the one in assignment 1.

The autoencoder model described above allows us to predict that "not bad" is actually neutral or positive, while the bag-of-vectors model would predict a more negative class.

(b) (5 points) How do the reconstructed vectors and reconstruction loss help in learning the parent representation?

The reconstruction loss prevents the network from ignoring the contribution of one of x_1 or x_2. In other words, it forces the network to capture the joint 'meaning' of x_1 and x_2.

(c) (9 points) Compute the following gradients for the reconstruction loss:
Hint: You can use the following notation:

$$1\{x > 0\} = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases}$$

Using it on a matrix would perform an element-wise operation, e.g.

$$1\{\begin{bmatrix} 1 & 0 \\ -1 & 3 \end{bmatrix} > 0\} = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}$$

i. (3 points) $\delta_1 = \frac{\partial J_1}{\partial p}$

$\delta_1 = \frac{\partial J_1}{\partial p} = W_2^T(x' - x)$

ii. (3 points) $\delta_2 = \frac{\partial J_1}{\partial h}$ (where $h = W_1 x + b_1$)

$\delta_2 = \frac{\partial J_1}{\partial h} = \delta_1 \cdot 1\{h > 0\}$

iii. (3 points) $\frac{\partial J_1}{\partial W1}$

$\frac{\partial J_1}{\partial W1} = \delta_2 x^T$

3. Vanishing Gradients (15 points)

(a) (15 points) ___ suffer(s) from the vanishing gradient problem. Circle all that apply and JUSTIFY YOUR ANSWER.

i. 1-Layer Feed-forward NN
ii. Very Deep Feed-forward NN
iii. Recurrent NN
iv. Word2vec CBOW
v. Word2vec Skip-Gram

Very Deep Feed-forward NN, Recurrent NN. All of these models are "deep", involving chained matrix multiplication and possibly saturating non-linearities.

4. BERT (10 points)

(a) (5 points) For the same task as previous question, would using BERT embeddings instead of Word2Vec provide any benefit? Explain why. Provide a concrete example(with actual embeddings). Make assumptions as needed.

Yes. While word2vec does embed meaning in the vectors, but the embedding for each vector is the same independent of the context it appears in. For example, the word **ridiculous** generally has a negative sentiment. But in this sentence: **the movie was ridiculously good** it has a positive sentiment. For example while, word2vec may give this word a vector like $[0.1, 0.2, 0.2, ...]$, BERT would give it a different vector based on the context it appears in ($[0.9, 0.8, 0.7, ...]$) and its vector would be different ($[0.2, 0.3, 0.1, ...]$) when it appears in a sentence like **it was a ridiculous movie.**

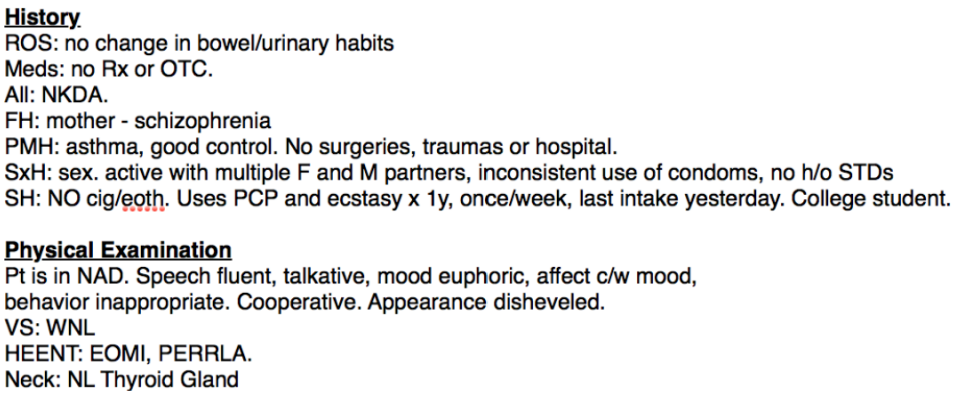
(b) (5 points) Explain what the purpose of the [CLS] token in BERT.

This token was intended to be used for classification of the entire sentence.

5. Feature Engineereing (36 points)

You are consulting for a healthcare company. They provide you with clinical notes of the first encounter that each patient had with their doctor regarding a particular medical episode. There are a total of 12 million patients and clinical notes. Figure 2 shows a sample clinical note. At the time that each clinical note was written, the underlying illnesses associated with the medical episode were unknown to the doctor. The company provides you with the true set of illnesses associated with each medical episode and asks you to build a model that can infer these underlying illnesses using only the current clinical note and all previous clinical notes belonging to the patient. The set of notes provided to you span 10 years; each patient therefore can have multiple clinical notes (medical episodes) in that period. Each note can contain any number of tokens (see Figure 2). Some tokens (e.g. "Meds") occur more frequently than others in the collection of notes provided to you. You call your Professor for advice. He tells you to first create a distributed representation of each patient note by combining the distributed representations of the words contained in the note.

(a) (6 points) Given the sample note provided in Figure 2, how would you map the various tokens into a distributed vector representation?



History
ROS: no change in bowel/urinary habits
Meds: no Rx or OTC.
All: NKDA.
FH: mother - schizophrenia
PMH: asthma, good control. No surgeries, traumas or hospital.
SxH: sex. active with multiple F and M partners, inconsistent use of condoms, no h/o STDs
SH: NO cig/eoth. Uses PCP and ecstasy x 1y, once/week, last intake yesterday. College student.

Physical Examination
Pt is in NAD. Speech fluent, talkative, mood euphoric, affect c/w mood,
behavior inappropriate. Cooperative. Appearance disheveled.
VS: WNL
HEENT: EOMI, PERRLA.
Neck: NL Thyroid Gland

Figure 2: Sample clinical note (source USMLE)

Given that there is no proper sentence structure a bag of words approach would be most appropriate. We awarded full credit to students who recommended other word vector representation techniques (Skip-gram, GloVe, CBOW etc.). We awarded partial credit to students who did not explicit discuss which word vector representation technique should be applied.

(b) (5 points) You have the option of representing each note as the summation of its constituent word vectors or as the average of its word vectors. Both seem reasonable. What's your best course of action? Briefly justify your selection.

Try both and see which one yields better results downstream. We awarded full credit to students who proposed either averaging or summing and provided a reasonable justification for their choice (e.g. summing captures the length of the note, averaging disregards the length etc.)

(c) (5 points) Your Professor tells you that you must normalize (magnitude-wise) your wordvectors before you perform the operation you decided to do in 2). Assuming you might try a standard neural network model, for which nonlinearities might that matter more?

It will matter for non-linearities that flatten in higher regimes (softmax, sigmoid, tanh etc.). It will not matter for RELU. Large-magnitude word vectors will cause the non-linearity units to saturate. At saturation regimes the gradient is close to zero. Your model will not learn effectively as the error would not propagate. Partial credit was awarded to students who identified the non-linearities for which normalization is important, but did not provide a suitable justification thereof.

(d) (5 points) A patient may have any number of illnesses from a list of 70,000 known medical illnesses. The output of your recurrent neural network will therefore be a vector with 70,000 elements. Each element in this output vector represents the probability that the patient has the illness that maps to that particular element. Your Professor tells you that illnesses are not mutually exclusive i.e. having one illness does not preclude you from having any other illnesses. Given this insight, is it better to have a sigmoid non-linearity or a softmax non-linearity as your output unit? Why?

Sigmoid. In the softmax case, the presence of one disease would lower the probability of all other diseases. This contradicts our assumption that the diseases are not mutually exclusive.

(e) (5 points) You try to figure out a better way to reduce the training and testing time of your model. You perform a run time analysis and observe that the computational bottleneck is in your output unit: the number of target illnesses is too high. Your Professor tells you that each illness in the list of 70,000 illnesses belongs to one of 300 classes (e.g. a migraine belongs to the neurological disorder class). He shares with you a dictionary which maps each illness to its corresponding class. How can you use this information to reduce the time complexity of your model? Include your forward propagation equations in your answer.

Use the last hidden layer to predict the class of the disease. Take the K top classes and predict the actual disease from the same last hidden layer. In this way you are only ever predicting the relevant subset of the full disease distribution given your hidden layer with minimal loss of accuracy. You are also propagating your errors from both outputs to the same hidden unit. We awarded partial credit to students that said to predict the classes instead of the diseases. This approach would lead to a loss of accuracy.

$$c_1 = sigmoid(\Sigma_j h_j(t) w_{1j})$$
$$y_c = sigmoid(\Sigma_j h_j(t) v_{cj})$$

(f) (5 points) Your model now trains (and predicts) several times faster. You achieve a precision score of 72% on positive cases (true positives) and a precision score of 68% on negative cases (true negatives). Confident with your initial results, you decide to make a more complex model. You implement a bidirectional deep recurrent neural network over the chronologically ordered patient note-vectors. Your new results are stellar. Your positive precision is 95% and your negative precision is 92%. You boast to your Professor that you have built an AI doctor. You coin the name Dr. AI for your model. Unfortunately, your Professor tells you that you have made a mistake. What is the mistake?

Leakage. You are using future information to predict the current disease. We were generous with this question, and awarded full credit to students that argued that a measurement of recall was needed. We awarded partial credit to students that claimed the the network might have been over-fitting.

(g) (5 points) Which level(s) of the linguistic hierarchy were you tackling here?

Semantic.