Your Name: _____

# COMP 394 (NLP), Fall 2024

## Practice Exam 2

Suhas Arehalli

**Notes about this exam:**

- Please write legibly and large enough so I can read your answers.
- You **may and should** ask me for clarification if you have any questions.
- You may use your handwritten notecard you have created.
- This is an individual assignment **DO NOT** discuss it with anyone.

**Classifiers**

Suppose you've trained a binary logistic regression model to predict whether a sentence was written by author A (label -1) or author B (label 1). You have hand-designed the following set of features: $f_0$: the presence of punctuation (0 or 1), $f_1$: the length of the sentence in words, $f_2$: the number of times the word "dog" appears in the sentence, and $f_3$: the number of times the word "cassowary" appears in the sentence.

Note that we feed the input into the model as the feature vector

$$x = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{bmatrix}.$$

You train the model with SGD (as we discussed in class) and we learn the weight vector

$$w = \begin{bmatrix} -3.20 \\ 0.35 \\ 1.05 \\ -0.35 \end{bmatrix}$$

(a) Is the input *the dog and the cassowary ran away* more likely to be written by author A or B?

**Solution:**

$$\begin{aligned} f_0 &= 0 \\ f_1 &= 7 \\ f_2 &= 1 \\ f_3 &= 1 \\ w \cdot x &= 0.35 \cdot 7 + 1.05 - 0.35 \\ &= 2.45 + 1.05 - 0.35 \\ &= 3.15 > 0 \end{aligned}$$

So label 1 is more likely. This is because $\sigma(x) > 0.5$ for $x > 0$ — no more info about the sigmoid function is necessary!

(b) According to your model, does the presence of punctuation (independent of other features) make a sentence more likely to be written by author A or B? Are longer sentences more likely to be written by author A or B? Which author is more likely to be identified by the use of the word *dog*? What about *cassowary*?

**Solution:** Look at the sign! A positive sign means the larger the feature, the more likely it is the label should be 1.

**Word Embeddings**

(a) Which of the following are likely to be *sparse* vector representations? Circle the letters corresponding to these choices.

a. TF-IDF document vectors
b. word2vec word vectors
c. PPMI word vectors
d. a one-hot vector

**Solution:** Sparsity/Density refers to how many dimensions will carry information/be non-zero — Sparse vectors have fewer dimensions that carry information, dense vectors have more. Sparse vectors would those in TF-IDF (most words will not be in most documents), PPMI (most words will no co-occur with eachother), and one-hot vectors (maximally sparse by definition). Of course, these vectors can be made dense by applying some sort of dimensionality reduction, which the textbook mentions, but we only need to do that because they are naturally sparse!

(b) Which of the following most accurately describes the skip-gram task?

a. Predict a word's meaning given the word
b. Predict a target word given surrounding context words.
c. Predict context words given a target word.
d. Predict a label given a sequence of tokens.

**Solution:** c is correct: We predict whether a context word is a real or fake context word based on an n-gram window around a target word in the training corpus. b (the reverse task) is CBOW, the other word2vec task.

(c) Briefly describe what is meant by the *distributional hypothesis* and how it motivates the training of dense word vectors.

**Solution:** The distributional hypothesis states that a word's meaning can be determined by it's *distribution* — the contexts in which it appears. This is why our word-vector training algorithms use co-occurrence as their training objective rather than an explicit task based on meaning (i.e., like option a in the previous question).

**Feedforward NNs**

A colleague is building a sentiment analysis model on movie reviews, and they begin by using the simplest model you can think of — a perceptron trained on hand constructed features.

(a) They run into a problem with 2 specific features: One indexes the presence of the word *don't* and the other the presence of the word *not*. These features are such that you classify the sentences that have one of these features correctly, but the model makes errors when both are activated.

To help them diagnose the issue, you look at some of the misclassified examples and see the sentence *I don't mean it's not good — it is!*.

**What does this problem indicate? How might you modify the model to make it work?** Briefly defend your suggestion, using the language and examples we've developed in class.

**Solution:** This is a direct parallel to the XOR problem: we want a representation where the representation our model learns can handle double negation: The sentence is negated if only one of the two described features is activated. The XOR problem tells us that a perceptron cannot learn a solution to this problem, and so a solution may suggest moving to a more complex model (a feedforward NN) or hand-engineering a more complex feature (taking the XOR of these two features).

(b) Your colleague later decides that they need a model that allows for neutral labels. They approach this by training a model that produces 3 scores, one for each label, that indicates how well the input matches that label. For now, they are classifying based on which model gets the highest score, but they would like to also provide a probability distribution over the 3 labels for each input.

**What would you suggest they do? What tools might be helpful to them?**

**Solution:** The standard way of converting scores to a probability distribution is *softmax*! Partial (i.e., the majority of) credit would be given to some other kind of normalization approach, though they should be careful to handle negative scores (just dividing by the sum would result in negative probabilities!)

**Modern NN Architectures**

(a) We briefly discussed how n-gram models can fail to capture phenomena like agreement. For example, a 4-gram models cannot, in principle, reliably predict the number of the verb that can come after *the dog near the gardens.*. You can construct such an example for an n-gram model for any choice of $n$, including a neural n-gram model.

**Briefly explain why you cannot construct such in-principle counterexamples for an RNN model.**

> **Solution:** An RNN model allows us to have a theoretically unbounded context! In practice, we face issues like the vanishing gradient problem, etc. that may cause in-practice failures and training practices that concede to computational limitations, but there is nothing in-principle preventing the model from learning a dependency of any length in the input.

(b) Is the same true for a transformer model? Explain.

> **Solution:** Transformers, while being very popular, do NOT solve this issue. A standard transformer maintains a fixed context-window over the input over which we compute attention, and so a dependency that is longer than that context window will be ignored. However, it's technically possible to compute attention over sequences of varying length, so if you discuss that it's possible to build an unbounded context transformer, despite being especially computationally expensive (a practical issue!), I'd give full points.
>
> Note that this ambiguity is why this is a practice question and not an exam question!