**Part I: Language modelling**

**Q1. Entropy**

*(See: [Goldsmith], p.17)*

Take the following two sentences as a mini-corpus.

<start> how much wood would a woodchuck chuck if a woodchuck could chuck wood a woodchuck would chuck as much wood as a woodchuck could chuck if a woodchuck could chuck wood <end>

**Q1a -**Compute the Shannon entropy for the distribution of unigrams and bigrams in this text. Which distribution has more entropy and why?

**Sol:**

**Entropy of unigrams: 3.161170216016749**

**Entropy of bigrams: 6.89287868934203**

**Bigrams has more entropy as it’s easy to determine the probability of next word taking context or the previous word into account.**

**Q1b** - How would the entropy of these distributions change if you add +1 Laplacian smoothing?

**Sol:**

**Entropy of unigrams after smoothing: 3.2317345534943884**

**Entropy of bigrams after smoothing: 8.162960208752768**

**Q2. Perplexity**

*(See: [JMv2] p.95 or [JMv3] p. 59)*

<start> would a woodchuck chuck wood if it could chuck wood <end>

<start> wood a woodchuck chuck would if it could chuck would <end>

**Q2a -**Compute the (length-normalized) perplexity of the two sentences above using unigram and bigram language models "trained" on the mini corpus from Q1 (so, four scores total)**.***Note: you will have to use smoothing to account unforseen ngrams*.

**Q2b -**What do these perplexity scores tell you about the unigram vs. bigram language models you computed?

**Sol:**

Perplexity of bigram of sentence 1 over trained corpus is: 7.537890760509168

Perplexity of bigram of sentence 2 over trained corpus is: 10.064149740666245

Perplexity of unigram of sentence 1 over trained corpus is: 6.868344257764572

Perplexity of unigram of sentence 2 over trained corpus is: 5.206311147629566

Usually the perplexity of unigrams is more compared to bigrams. Here considering the training and test corpus and the size of vocabulary the perplexity scores of unigrams are less than the bigrams.

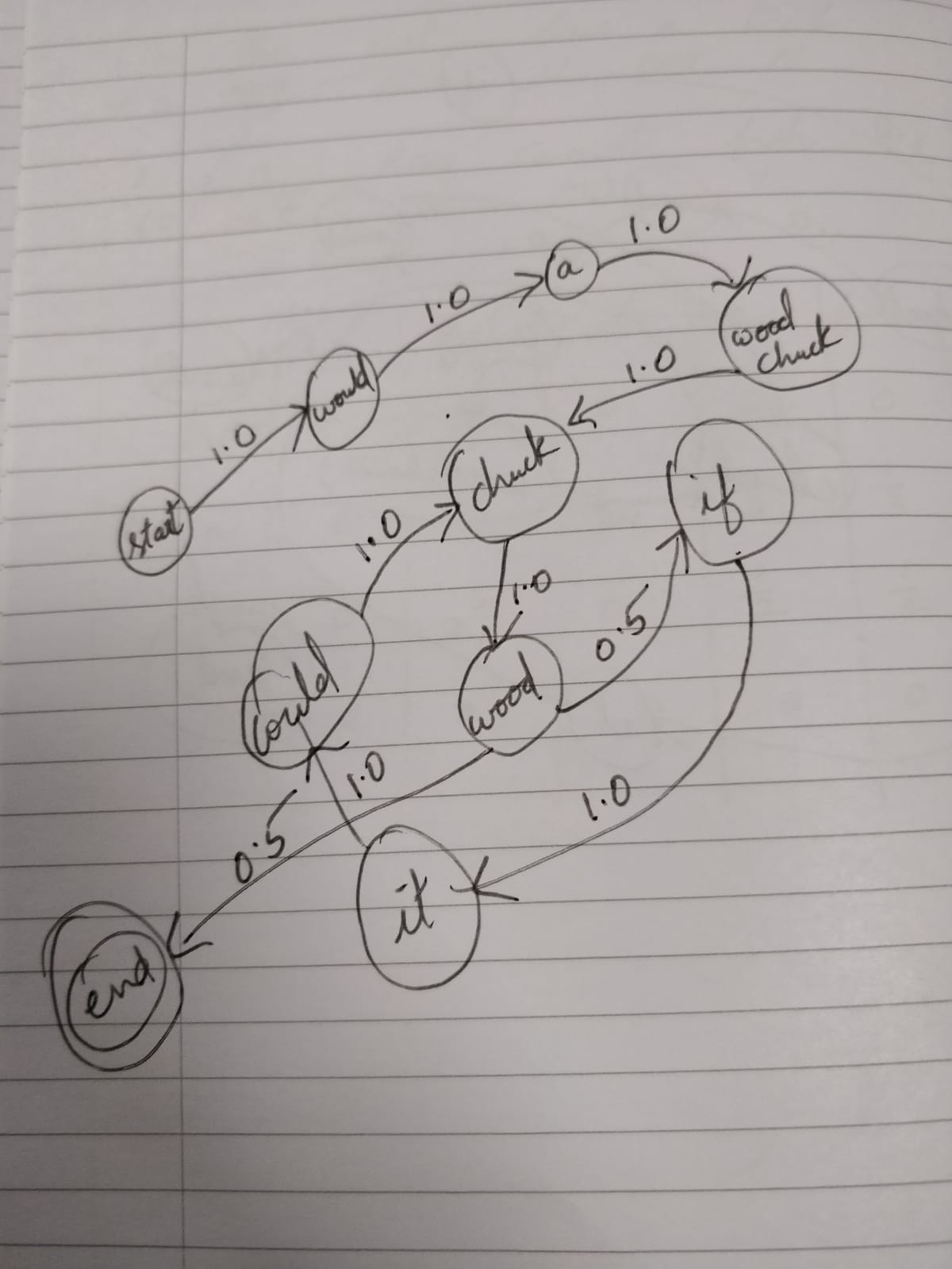
Usually its not the same for corpus with large set of unique tokens.

***Q3. Bigram PFST***

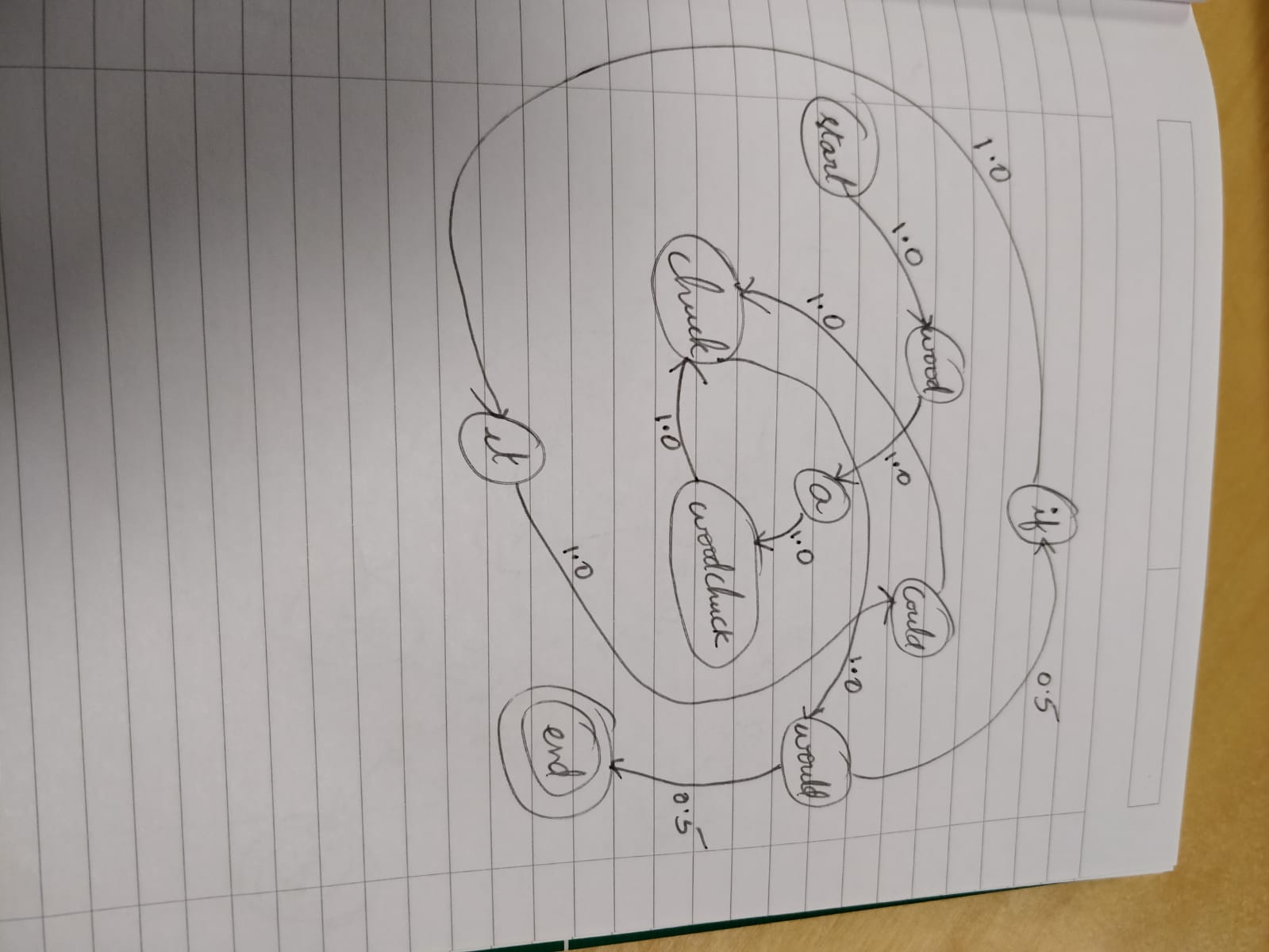
*(See: Lecture 4 slides)*

**Q3*-***Diagram a probabilistic finite state transducer representing the bigram language model from Q2.

<start> would a woodchuck chuck wood if it could chuck wood <end>



<start> wood a woodchuck chuck would if it could chuck would <end>



**Part II: Part of Speech Tagging**

*(See: [JMv2] chapter 5)*

**Q4: Types of taggers**

**Q4a -** One of the simplest kinds of taggers is a lookup tagger that simply maps each word to a single part of speech. It is easy to implement, but what are the disadvantages of such a tagger? Mention 2.

**Sol:**

1. **Does not apply any rules to tag a word**
2. **Does not take context into account**
3. **Does not incorporate frequency or probability of word with particular tag**
4. **Does not consider any of the tags associated to previous words**

**Q4b -** Suppose we want to build a better tagger for English which uses word affixes to determine the PoS of a word (e.g. plural nouns generally end in *s*). Pick 3 tags from the Penn tagset and write a regular expression for each. Can you think of any words which will be incorrectly tagged (false positives or false negatives) by your tagger?

**Sol:**

1. **(r’.\*ing$’, ‘VBG’)**

**thing, icing, wing,**

1. **(r'.\*ment$', 'NN')**

**Argument,** **arrangement, judgement, deployment**

1. **(r'.\*ed$', 'VBD')**

**simple-minded, guided, stored**

1. **(r’.\*s$’, ‘NNS’)**

**Mans,kiss,stress**

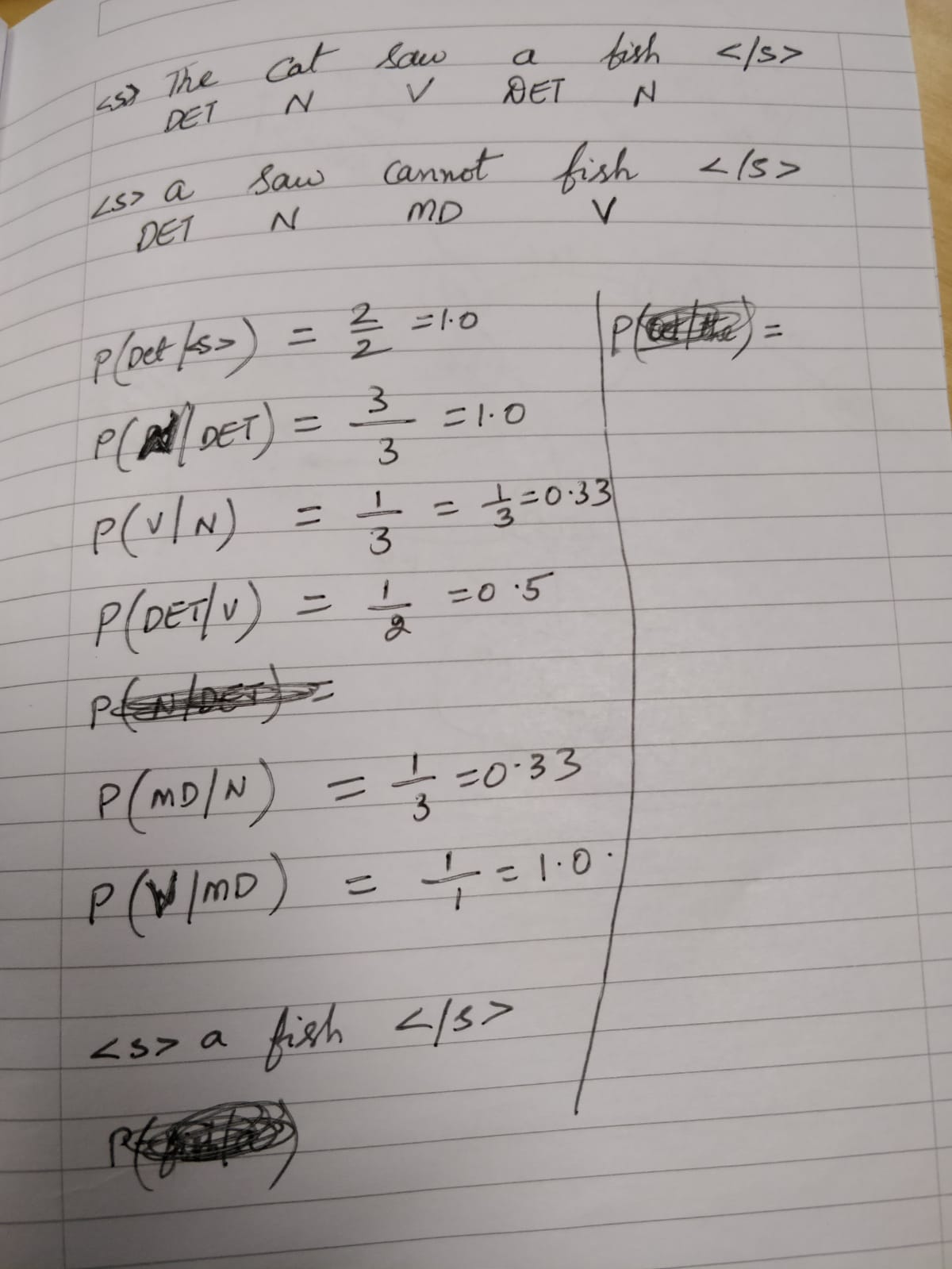
**Q4c -** Neither of the taggers above take context into account. Assume that you then build a bigram HMM tagger from the following two-sentence corpus:

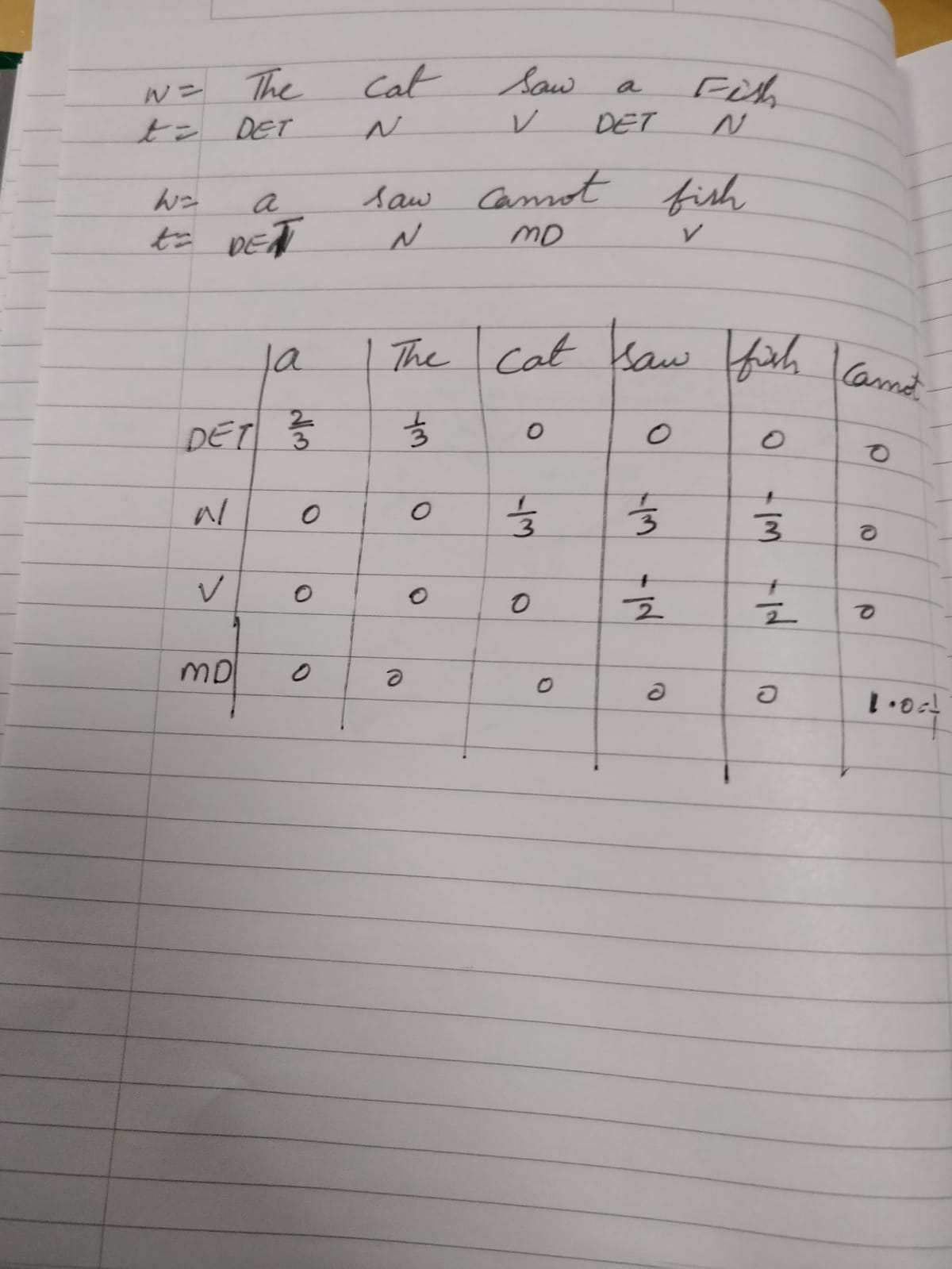
* the/DET cat/N saw/V a/DET fish/N
* a/DET saw/N cannot/MD fish/V

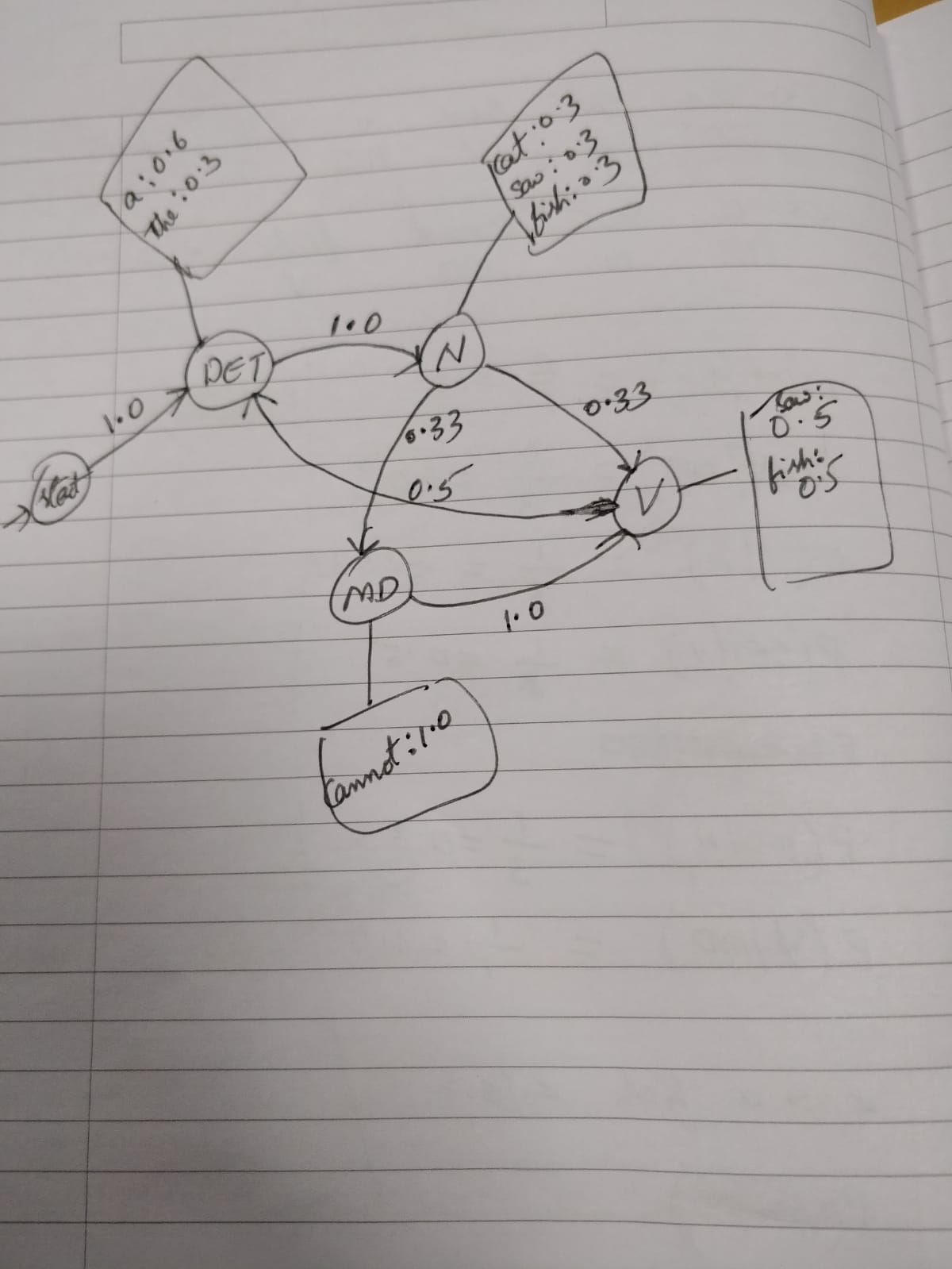
Your tagger is now given the phrase "a fish". How will it tag the word *fish* in this phrase, and why? Show your calculations.

**Sol:**

**‘a fish’ is tagged as ‘a/DET fish/N’**







**Q5: Playing with the Core NLP tagger**

The Stanford CoreNLP tagger uses the [Penn Treebank tagset](https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html). You can play around with it [here.](http://corenlp.run/)

**Q5a -** Find a sentence where this (roughly state-of-the-art) tagger tags at least one word incorrectly. What's your explanation for why it got it wrong? For hints, take a look at [this paper.](https://nlp.stanford.edu/pubs/CICLing2011-manning-tagging.pdf)

Sol:

**I think I can open the Can, even though this Can is difficult to open**

Here ‘Can’ is noun but tagged as ‘MD’.

Because can and can are similar here.

**Q5b -**Think of a downstream application that might be impacted by the mistake you found. What could go wrong?

While using a language translator this causes ambiguity to understand by the machine if ‘can’ is ‘MD’ or ‘Noun’ and changes the meaning in other language.