Entropy - Language Models

Use the three corpora included in directory corpus/:

- en.txt A fragment of EFE corpus in English
- es.txt A fragment of EFE corpus in Spanish
- taggedBrown.txt A fragment of Brown corpus in English Pos-tagged

and a set of python functions in auxiliar.py:

- getWordsFromFile(inF):
- "get a list of words from a text file"
- getTaggedWordsFromFile(inF):
- "get a list of pairs <word,POS> from a text file"
- getTagsFromTaggedWords(l):
- "from a list of tagged words build a list of tags"
- countNgrams(l,inic,end=0):
- From a list l (of words or pos), an inic position and an end position
- a tuple(U,B,T) of dics corresponding to unigrams, bigrams and trigrams
- is built

to answer to the following questions:

1. write a python function for computing the order 0 (unigram) model of en.txt

$$H = -\sum_x p(x)\, log\ p(x)$$

2. write a python function for computing the order 1 (bigram) model of en.txt

$$H = -\sum_x p(x) \sum_y p(y|x)\, log\ p(y|x)$$

3. write a python function for computing the order 2 (trigram) model of en.txt

$$H = -\sum_x p(x) \sum_y p(y|x) \sum_z p(z|xy)\, log\ p(z|xy)$$

4. Use now the taggedBrown.txt corpus. Compute the perplexity of the trigram language model for three different sizes of the corpus (the full corpus, half of it and a quarter of it).
5. Smooth the trigram language model going from the trigrams <x,y,z>, to <x',y,z> and to <x',y',z>, where x' is the POS of x and y' is the POS of y. Compute the perplexity as in the previous case. Build the following table and discuss the results.

perplexities	Full corpus	½ corpus	¼ corpus
<x,y,z>			
<x',y,z>			
<x',y',z>			