Deep Structured Learning (IST, Fall 2020)

Homework 2

Instructor: André Martins
TA: Marcos Treviso

Deadline: Wednesday, November 18, 2020.

Please turn in the answers to the questions below, **in English**, together with the code you implemented to solve them (when applicable). Please email your solutions in **electronic format** (a single zip file) with the subject "Homework 2" to:

`deep-structured-learning-instructors@googlegroups.com`

Hard copies will not be accepted.

Question 1

Dynamic Programming. Your friend John the Dynamic lives in Lisbon and, depending on the weather conditions, he enjoys surfing, going to the beach, playing video games, and studying machine learning. His activities are governed by a hidden Markov model, where the hidden variables correspond to the weather (`Sunny`, `Windy`, and `Rainy`) and the observed variables correspond to `Surf`, `Beach`, `Videogame`, and `Study`. The emission and transition probabilities are shown in Tables 1–2.

	`Sunny`	`Windy`	`Rainy`
`Surf`	0.4	0.5	0.1
`Beach`	0.4	0.1	0.1
`Video game`	0.1	0.2	0.3
`Study`	0.1	0.2	0.5

Table 1: Emission probabilities: rows conditioned on columns.

	`Sunny`	`Windy`	`Rainy`
`Sunny`	0.6	0.3	0.2
`Windy`	0.3	0.5	0.3
`Rainy`	0.1	0.2	0.5

Table 2: Transition probabilities: rows (timestep $t + 1$) conditioned on columns (timestep t).

1. John's activities for the past week were like shown in Table 3. **Assume that the weather on October 7 was rainy, and on October 15 it was sunny.** In class we saw two dynamic programming algorithms: the forward-backward algorithm (for which we provide an implementation in `hw2_decoder.py`) and the Viterbi algorithm (which you may need to implement as part of this exercise).

Monday, Oct 8	`Videogame`
Tuesday, Oct 9	`Study`
Wednesday, Oct 10	`Study`
Thursday, Oct 11	`Surf`
Friday, Oct 12	`Beach`
Saturday, Oct 13	`Videogame`
Sunday, Oct 14	`Beach`

Table 3: John's activities for the past week.

(a) (15 points) Knowing John's activities, what was the most likely weather for the past week? Which algorithm did you use to answer this question?

(b) (10 points) Let's suppose you made a bet with John where you receive 1€ for every day you guess the weather correctly, and you lose 1€ if your prediction is wrong. As in the previous question, you observed John's activities and the weather in October 7 and October 15. What would be your bet for the weather in the days from October 8 to 14? Which algorithm maximizes your expected profit?

2. (5 points) Actually, John never surfs two days in a row because (despite his nickname) he gets exhausted and he needs to rest at least one day before going back to the water. Can we accommodate this extra piece of knowledge in a hidden Markov model? Justify.

Question 2

Sequential OCR. So far, all your OCR experiments used models that try to predict each character independently from the others. In this exercise, you will solve the problem with structured prediction, using a linear sequential model implemented in PyTorch.

1. Exploiting the sequential structure of the characters (as they form words), implement the `compute_scores`, `compute_log_partition` and `viterbi` in the `LinearCRF` class in the provided file `hw2_linear_crf.py`. Notice that the first two methods are needed for the loss calculation, whereas the last is needed for decoding. Use a batch size of 1 if you wouldn't like to deal with padding & masking. The skeleton code contains an additional `mask` variable, which you can use in case you want to batch your data.

   Hint: follow the implementations of Viterbi and Forward-Backward from the previous exercise, using the `forward` algorithm to compute the log-partition function. Note that the `autograd` toolkit automatically handles the computation of the `backward`[1] algorithm. See Jason Eisner's tutorial paper here: `https://www.aclweb.org/anthology/W16-5901.pdf`

   (a) (20 points) As unigram features, use the raw binary pixels. As bigram features, use a constant feature $\phi_{i,i-1}(x) = [1]$, i.e., the model will learn a weight for the conjunction of the two consecutive labels with no dependency on the pixels (i.e., a total of 26^2 weights for the bigrams). How does test accuracy compare with not using any structure?

   (b) (10 points) Repeat (a) with the pairwise features for the unigrams you used in homework 1 (keep the bigram feature unchanged).

[1] Not to confuse with Pytorch's backward method.

Question 3

Sequential OCR with RNNs. In the previous exercise, you solved the OCR problem with structured prediction, using a linear sequential model. You will now try out a recurrent neural network, more precisely a BILSTM tagger.

1. (25 points) Exploit the sequential structure of the characters (as they form words). Use as input a feedforward layer shared by all characters, followed by a bidirectional LSTM (BILSTM). Then, append an affine transformation followed by an output softmax layer to map this to character predictions. Use the raw binary pixels as input features. Regularize and optimize at your will. Plot the training loss and validation accuracies over epoch number. Report the final test accuracies. Hint: if you're using Pytorch, use the function `nn.LSTM` for this exercise.

2. (5 points) What are the advantages and disadvantages of this BILSTM compared to the CRF you implemented in the previous exercise?

3. (10 points) Repeat the previous question by appending your LinearCRF layer on top of the BILSTM tagger.