Prof. Ryan Cotterell

Assignment 3: Finite-State Automata

19/11/2022 - 08:18h

The first half of the assignment is a series of theoretical questions about the material. When you have answered the questions to your satisfaction, you should upload a pdf file with your solutions (preferably written in LaTeX) to Moodle. The second question is a coding task that is to be solved in the released Google Colab notebook. You have to copy the notebook to your own drive in order to edit it. When submitting your solution, please execute all cells in your copied notebook, then save it and include a link to your notebook in your PDF file submission. We will run software on your submission to *test for plagiarism.*

Important: Please ensure, that all cells in the notebook are already **executed** and that the notebook is accessible via the shared link in **Editor mode**! The notebook must also be executable from top to bottom without throwing errors.

Question 1: Exploring the Kleene Star (15 pts)

The goal of this question is to get you used to the idea of a Kleene star.

Definition 1. *One definition of a closed semiring is as follows. A semiring $\mathcal{W} = \langle A, \oplus, \otimes, \mathbf{0}, \mathbf{1}\rangle$ is called **closed** if the following two properties hold [1].*

- *For any countable sequence of elements $\{a_n\} \subset A$, we have that $\lim_{N\to\infty} \bigoplus_{n=1}^{N} a_n$ is well defined, in A, and invariant to the ordering of the elements of the sequence a_n;*
- *Second, the distributive property holds for infinite sums and products.*

a) **(3 pts)** Under Definition 1, show that $a^* \overset{\text{def}}{=} \bigoplus_{n=0}^{\infty} a^{\otimes n}$ satisfies the following equation

$$a^* = 1 \oplus a \otimes a^*. \tag{1}$$

b) **(3 pts)** Let $\mathcal{W}_{\log} = \langle \mathbb{R} \cup \{-\infty\}, \oplus_{\log}, +, -\infty, 0\rangle$ be the **log semiring**, where $\oplus_{\log}$ is defined as $a \oplus_{\log} b \overset{\text{def}}{=} \log\left(e^a + e^b\right)$. Find a Kleene star for $\mathcal{W}_{\log}$.

c) **(3 pts)** Find a Kleene star for the expectation semiring defined in Assignment 2.

d) **(6 pts)** Let $\mathcal{W}_{\text{lang}} = \langle 2^{\Sigma^*}, \bigcup, \otimes, \{\}, \{\varepsilon\}\rangle$ be the **language semiring**, where $\otimes$ is defined as $A \otimes B \overset{\text{def}}{=} \{a \circ b \mid a \in A, b \in B\}$ with $\circ$ as string concatenation. Note that 2^{Σ^*} is the powerset of Σ^*, i.e., the set of all possible languages. Prove $\mathcal{W}_{\text{lang}}$ is a semiring (check the semiring axioms), and find a Kleene star for $\mathcal{W}_{\text{lang}}$.

Question 2: Asterating Matrices in Idempotent Semirings (35 pts)

In this question, we will explore an algorithm for computing the asteration (Kleene star) of a matrix that is simpler (and slower) than Lehmann's. Despite those limitations, it is much easier to implement and often useful if you need a sanity check that your implementation of Lehmann's is correct. Rather than working with WFSA notation, we will consider asterating the matrix associated with a semiring-weighted graph. We write $\mathcal{G}$ for a weighted directed graph with N nodes and $\boldsymbol{M} \in \mathbb{R}^{N\times N}$ its adjacency matrix.[1] We have that $M_{ik} = w_{ik}$ encodes the weight w_{ik} on the edge $i \xrightarrow{w_{ik}} k$.

Definition 2 (Tropical Semiring)**.** *The semiring $\langle \mathbb{R}_{\geq 0}, \min, +, \infty, 0\rangle$ is called the **tropical semiring**.*

Definition 3 (Arctic Semiring)**.** *The semiring $\langle \mathbb{R}_{\leq 0}, \max, +, -\infty, 0\rangle$ is called the **arctic semiring**.*

Definition 4 (0-closed Semirings)**.** *A semiring $\langle A, \oplus, \otimes, \mathbf{0}, \mathbf{1}\rangle$ is called 0**-closed** iff, for every $a \in A$, we have $\mathbf{1} \oplus a = \mathbf{1}$.*

a) **(2 pts)** Prove the tropical semiring and the arctic semiring are 0-closed semirings.

b) **(4 pts)** Show that $\boldsymbol{M}^n \stackrel{\text{def}}{=} \overbrace{\boldsymbol{M} \otimes \cdots \otimes \boldsymbol{M}}^{n \text{ times}}$ encodes the sum of paths of length exactly n, i.e., that $(\mathbf{M}^n)_{ik}$ contains the semiring-sum of all paths from the node i to node k in graph $\mathcal{G}$ of length exactly n.

Definition 5 (Shortest Path Weight)**.** *Let $\mathcal{G}$ be a semiring-weighted graph. We denote with $\Pi(i,j)$ the set of all paths from the node i to the node j. The **weight of the shortest path** between i and j is defined as*

$$Z(i,j) \stackrel{\text{def}}{=} \bigoplus_{\boldsymbol{\pi} \in \Pi(i,j)} w(\boldsymbol{\pi}). \tag{2}$$

Here, $w(\boldsymbol{\pi}) = \bigotimes_{k=1}^{N+1} w_k$ denotes the **weight** of the path $\boldsymbol{\pi} = q_0 \xrightarrow{a_1/w_1} q_1 \cdots q_N \xrightarrow{a_{N+1}/w_{N+1}} q_{N+1}$. The shortest path weight in $\mathcal{G}$ is simply the sum over *all* weights in $\mathcal{G}$—this is exactly the pathsum!

The "shortest path weight" therefore refers to the *sum* over the path weights! While this might seem confusing at first, it is easy to see where the terminology comes from. If the semiring from which the weights come is the tropical semiring, the sum in eq. (2) turns into a min and the multiplication in the computation of the path weight turns into a sum. In the tropical semiring, the pathsum computes the weights of the shortest paths!

c) **(4 pts)** Show that, if the graph $\mathcal{G}$ with the transition matrix $\boldsymbol{M}$ is over a 0-closed semiring, the shortest path weight in eq. (2) depends only on paths of length of at most $N-1$ (i.e., the paths traversing at most $N-1$ transitions).

[1] To see how focusing on a single matrix $\boldsymbol{M}$ is without loss of generality, consider $\overline{\Sigma} = \Sigma \cup \{\varepsilon\}$ for some alphabet Σ. Then, we can define $\boldsymbol{M} \stackrel{\text{def}}{=} \bigoplus_{\sigma \in \overline{\Sigma}} \boldsymbol{M}^{(\sigma)}$ where $\boldsymbol{M}^{(\sigma)}$ is the σ-specific transition matrix.

d) **(2 pts)** Using your answer to (b) and (c), prove that in a 0-closed semiring, we have

$$\boldsymbol{M}^* = \bigoplus_{n=0}^{N-1} \boldsymbol{M}^n \tag{3}$$

e) **(3 pts)** Use Eq. (3) to come up a simple algorithm for computing $\boldsymbol{M}^*$. Give a (one-line) proof of correctness and a big-O bound on its runtime.

Definition 6 (Idempotency)**.** *A semiring $\langle A, \oplus, \otimes, \mathbf{0}, \mathbf{1}\rangle$ is called **idempotent** iff, for every $a \in A$, we have $a \oplus a = a$.*

f) **(2 pts)** Show that every 0-closed semiring is idempotent.

g) **(4 pts)** Show that in an idempotnent semiring, it holds that $\bigoplus_{n=0}^{K} \boldsymbol{M}^n = (\mathbf{I} \oplus \boldsymbol{M})^K$.

h) **(4 pts)** For any integer $n \in \mathbb{Z}_{>0}$, prove that we can we can write

$$\boldsymbol{M}^n = \bigotimes_{k=0}^{\lfloor \log_2 n \rfloor} \boldsymbol{M}^{\alpha_k 2^k} \tag{4}$$

where each $\alpha_k \in \{0, 1\}$. Use Eq. (4), (f), and (g) to come up with an asymptotically faster algorithm for 0-closed semirings than the one you developed in the previous question.

Now, we turn to analyzing non-idempotent semirings. Suppose we are working in the real semiring.

Definition 7 (Real Semiring)**.** *The semiring $\langle \mathbb{R}, +, \times, 0, 1\rangle$ is called the **real semiring**.*

In such a case, we have that $\sum_{n=0}^{\infty} \boldsymbol{M}^n \neq \sum_{n=0}^{K} \boldsymbol{M}^n$ for any $K < \infty$. However, we are still able to derive a bound on the performance of the approximation. To do so, however, we will need a bit of linear algebra.

i) **(3 pts)** Let $\mathbf{A} \in \mathbb{R}^{N \times N}$ be a real matrix. The **operator norm** of $\mathbf{A}$ is defined as

$$||\mathbf{A}||_2 \overset{\text{def}}{=} \sup_{\boldsymbol{x} \neq 0} \frac{||\mathbf{A}\boldsymbol{x}||_2}{||\boldsymbol{x}||_2} \tag{5}$$

Prove that $||\mathbf{A}||_2 = \sigma_{\max}(\mathbf{A})$, i.e., the largest singular value of $\mathbf{A}$.

j) **(4 pts)** Fix $K < \infty$. Find a closed-form expression for the truncation error $||\mathbf{A}^* - \sum_{n=0}^{K} \mathbf{A}^n||_2$ in terms of $\sigma_{\max}(\mathbf{A})$. Find a condition on $\sigma_{\max}(\mathbf{A})$ such that $\sum_{n=0}^{K} \mathbf{A}^n \to \mathbf{A}^*$ as $K \to \infty$.

k) **(3 pts)** Using your answer to the previous question, give a big-O bound in terms of K for the truncation error $||\mathbf{A}^* - \sum_{n=0}^{K} \mathbf{A}^n||_2$. In your opinion, given your bound, is using a truncation a good approximation to asteration?

Question 3: Implementation of a "Neural" Transducer (50 pts)

... well, almost. Despite the name of the question, you will **not** train a neural transducer in this assignment. Coding a fast implementation (or fast enough to train in a reasonable time in practice) of a neural transducer in a library such as PyTorch, especially when using a semiring different than the real one, is a challenging task and is actually an open research question. Therefore, we will use the library `rayuela` to implement the transducer and walk you through the exact steps you would need when training a neural transducer.

You will implement a finite-state transducer (FST) that maps an input $\boldsymbol{x}$ to all appropriate outputs $\boldsymbol{y}$. A path through the transducer from an initial state to a final state represents an *alignment* of $\boldsymbol{x}$ to $\boldsymbol{y}$, where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the concatenation of the input and output symbols along this path. In general, two strings $\boldsymbol{x}$ and $\boldsymbol{y}$ can be aligned through many paths, via different edit sequences (you can imagine turning some string $\boldsymbol{x}$ into $\boldsymbol{y}$ by first *deleting* a number of symbols in $\boldsymbol{x}$ and then *inserting* appropriate new ones in $\boldsymbol{y}$, or directly *replacing* the symbols). Many existing datasets consist of pairs of input and output strings $(\boldsymbol{x}, \boldsymbol{y})$ only, thus the alignment needs to be treated as a *latent variable.*

So how do we take into account all these possible latent alignments? The secret lies in **transducer composition**, which creates more complex transducers from simpler ones. More precisely, we need to compose a transducer encoding of the input string, $T_{\boldsymbol{x}}$, with a handcrafted edit-distance transducer, F. The composition $G = T_{\boldsymbol{x}} \circ F$ yields a transducer that has only the paths from F with input $\boldsymbol{x}$. Composing G with a transducer encoding of the output string, $T_{\boldsymbol{y}}$, yields a transducer such that all paths have input $\boldsymbol{x}$ and output $\boldsymbol{y}$. This transducer can be used for computing $p(\boldsymbol{y}|\boldsymbol{x})$.

If all of this still seems abstract to you, don't worry–the prepared assignment will guide you through the implementation to make sure you understand everything!

We ask you to code the answer to the following subquestions in a Google Colab notebook: `https://colab.research.google.com/drive/144jysknc_FNqkbr-R7H9g7Jo3O4tIrMC?usp=sharing`.

a) **(4 pts)** Implement the **log semiring** in the marked location in the notebook by defining the unit elements, addition, multiplication, and the Kleene star operator.

b) **(4 pts)** Implement a function which encodes an input string as a transducer by filling the marked location in the notebook. **Hint:** The input and output symbols on the arcs should be characters in the given alphabet. You should define your transducer in the **log semiring** as it will be useful when computing the log-likelihood.

c) **(7 pts)** Construct an edit-distance transducer whose states encode the most recent *output* symbol. The machine should have 3 types of transitions:

   1. **insertions** (no symbol is read, a symbol is outputted),
   2. **deletions** (a symbol is read, no symbol is outputted), and
   3. **substitutions** (a symbol is read, a symbol is outputted).

   Fill in the function in the marked location in the notebook. You should define your transducer in the **log semiring** as it will be useful when computing the log-likelihood.

d) **(15 pts)** Implement transducer composition by filling in the function in the marked location in the notebook. We provide some test cases for composition here: `https://drive.google.com/file/d/1wlHbFChZFcBgEU8knAeaUIqE0W4EaJEC/view?usp=share_link`.

e) **(8 pts)** Implement Lehmann's algorithm in the marked location in the notebook. **Hint:** As mentioned in Question 2, you can debug your implementation of Lehmann's with the simpler algorithm you developed there.

f) **(4 pts)** Compute the normalizer of a transducer by filling in the function in the marked location in the notebook.

g) **(4 pts)** Compute the log-likelihood $\log p(\boldsymbol{y}|\boldsymbol{x})$ by filling in the function in the marked location in the notebook.

h) **(4 pts)** Compute the *weight* of the most probable output string $\boldsymbol{y}$ given an input string $\boldsymbol{x}$ by filling in the function in the marked location in the notebook. **Hint:** You should reuse Lehmann's algorithm in a different semiring.

References

[1] John G. Fletcher. A more general algorithm for computing closed semiring costs between vertices of a directed graph. *Commun. ACM*, 23(6):350–351, jun 1980.