Advanced Human Language Technologies
Exercises on features for log-linear models

Features for classification

Exercise 1.

Given a vocabulary V with $|V| = 1000$, and a word history $x = \langle x_1, \ldots, x_{i-1} \rangle$, and assuming the size of our alphabet is 26 letters, we define the following feature template for any word u and 4-letter suffix s:

$$\mathbf{f}_{u,s}(x, y) = \begin{cases} 1 & \text{if } y = u \text{ and } x_{i-1} \text{ ends with } s \\ 0 & \text{otherwise} \end{cases}$$

1. How many possible features are there in this model?
   (a) 1000×26^4
   (b) 26^4
   (c) 1000
   (d) 1000×1000
2. Now consider a training set of size $n = 10,000$. Which is a good upper bound on the number of features introduced by this training set?
   (a) 1000×26^4
   (b) 10000×26^4
   (c) 1000
   (d) 10000

Exercise 2.

Consider the label set $\mathcal{Y}$ consisting of part-of-speech tags with $|\mathcal{Y}| = 50$, and the set $\mathcal{X}$ consisting of histories of the form $\langle t_1, \ldots, t_i, w_1, \ldots, w_n, i \rangle$. Also assume we have a vocabulary V of size 1000. We define the following feature template for any word $u \in V$ and PoS-tag $t \in \mathcal{Y}$:

$$\mathbf{f}_{u,t}(x, y) = \begin{cases} 1 & \text{if } w_i = u \text{ and } y = t \\ 0 & \text{otherwise} \end{cases}$$

1. How many possible features are there in this model?
2. How many different features are introduced by the training set consisting of the following word/tag pairs ?
   `the/DT man/NN fishes/V for/ADP the/DT fishes/NN`

Exercise 3.

Consider the label set $\mathcal{Y} = \{cat, dog, rat, cow\}$ with three simple features:

$$\mathbf{f}_1(x, y) = \begin{cases} 1 & \text{if } x = \texttt{the} \text{ and } y \text{ ends with } \texttt{at} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_2(x, y) = \begin{cases} 1 & \text{if } x = \texttt{the} \text{ and } y \text{ starts with } \texttt{c} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_3(x, y) = \begin{cases} 1 & \text{if } x = \texttt{the} \text{ and } y \text{ has second letter } \texttt{o} \\ 0 & \text{otherwise} \end{cases}$$

Say we are given the weight vector $\langle w_1, w_2, w_3 \rangle = \langle 1, 2, 3 \rangle$.

1. What is the score of the following pairs?
   - $(x, y) = (\texttt{the}, \texttt{cat})$
   - $(x, y) = (\texttt{the}, \texttt{dog})$
   - $(x, y) = (\texttt{the}, \texttt{rat})$
   - $(x, y) = (\texttt{the}, \texttt{cow})$
2. What is the probability of each label for $x = \texttt{the}$?
   - $P(\texttt{cat}|\texttt{the}; w) =$??
   - $P(\texttt{dog}|\texttt{the}; w) =$??
   - $P(\texttt{rat}|\texttt{the}; w) =$??
   - $P(\texttt{cow}|\texttt{the}; w) =$??
3. What would be the probabilities if the weight vector was:
   - $\langle w_1, w_2, w_3 \rangle = \langle 0, 0, 0 \rangle$.
   - $\langle w_1, w_2, w_3 \rangle = \langle 3, 1, 1 \rangle$.

Exercise 4.

Think of named entities, like *Michael Jackson*, *Barcelona*, or *United Nations*. We can distinguish between different types of entities. For example, *Michael Jackson* is a PERSON, *Barcelona* is a LOCATION and *United Nations* is an ORGANIZATION. Like in all NLP problems, we will frequently encounter difficult cases, such as

```
[ Jack London ] went to Paris.
```

where *Jack London* is a PERSON rather than a LOCATION despite containing a well known location name. There are cases of ambiguity, such as

```
[ Spain ] won the championship.
```

where *Spain* is an ORGANIZATION because in this case it refers to a team or country, not a geographical location. Hence we would like to use linear classification methods that predict the most appropriate class. Let Y be the set of named entity classes (three in our example). In this exercise **we will assume that a previous process has identified the boundaries of named entities, and our goal is just to classify them**. We will make the following linear predictions

$$\text{ne\_type}(x_{1:n}, i, j) = \underset{y \in Y}{\text{argmax}}\, \mathbf{w} \cdot \mathbf{f}(x_{1:n}, i, j, y)$$

where

- $x_{1:n}$ is a sentence with n tokens (x_i is the i-th token)
- i and j are the first and last positions of the entity; thus $x_{i:j}$ is the complete entity

- y is one of the entity types
- $\mathbf{f}(x_{1:n}, i, j, y)$ is a function that returns a feature vector
- $\mathbf{w}$ is a vector of parameters, with one weight per feature

We are interested in creating a set of features for the classifier. For example, feature 1 may indicate if the current entity is the single-word entity *London* and is tagged as LOC (location):

$$\mathbf{f}_1(x_{1:n}, i, j, y) = \begin{cases} 1 & \text{if } i = j \text{ and } x_i = \texttt{London} \text{ and } y = \text{LOC} \\ 0 & \text{otherwise} \end{cases}$$

Rather than specifying each dimension explicitly, we will create *feature templates* that generate a number of features mechanically. Each feature template is identified by a type. For example, the template with type 1 may be

$$\mathbf{f}_{1,l,a}(x_{1:n}, i, j, y) = \begin{cases} 1 & \text{if } i = j \text{ and } x_i = a \text{ and } y = l \\ 0 & \text{otherwise} \end{cases}$$

and will generate an actual feature for every $l \in Y$ and every word $a \in V$ (where V is the set of words in our language). If we set $l = \text{LOC}$ and $a = \text{London}$ we obtain the feature above. In general, this template will create feature dimensions identified by a tuple of the form $\langle 1, l, a \rangle$. We will assume that the parameter vector $\mathbf{w}$ is indexed with the same type of tuple dimensions. For example, $\mathbf{w}_{1,\text{LOC,London}}$ is the parameter associated with the feature above.

4.1. Feature Templates

Write feature templates that capture the following information. For each template, specify the tuple that identifies the feature dimensions. Justify your answers if necessary

- Type 1: The entity is word a
  Answer:
  $$\mathbf{f}_{1,l,a}(x_{1:n}, i, j, y) = \begin{cases} 1 & \text{if } i = j \text{ and } x_i = a \text{ and } y = l \\ 0 & \text{otherwise} \end{cases}$$
- Type 2: All entity words are capitalized.
- Type 3: The entity contains word a.
- Type 4: The entity has at least three words, the first is a and the second is b.
- Type 5: Word a appears right before or right after the entity.

4.2. Feature Vectors

Assume we define a feature function with the templates 1 to 5 above. Assume also that our vocabulary V includes all English words. For each case below, write the names of the features that are non-zero:

1. $x_{1:5} = \texttt{I live in Barcelona .}$
   $\mathbf{f}(x_{1:5}, 4, 4, \text{LOC}) = \mathbf{f}_{1,\text{LOC,Barcelona}}, \ldots$
2. $x_{1:12} = \texttt{Smith wrote in the Journal of the Royal College of Physics .}$
   $\mathbf{f}(x_{1:12}, 1, 1, \text{PER}) = ?$
   $\mathbf{f}(x_{1:12}, 5, 11, \text{ORG}) = ?$
3. $x_{1:6} = \texttt{Spain won the World Cup .}$
   $\mathbf{f}(x_{1:6}, 1, 1, \text{ORG}) = ?$
   $\mathbf{f}(x_{1:6}, 1, 1, \text{LOC}) = ?$
   $\mathbf{f}(x_{1:6}, 4, 5, \text{ORG}) = ?$

4.3. Feature Weights

We will now play the role of a learning algorithm by setting the weights of features. The main point is to show that, while feature templates generate a large number of actual features, a learning algorithm should be able to select a few features which are highly discriminative.

Let's start with a simple example. Assume our training set has two examples:

`[ Barcelona ]`$_{\text{LOC}}$ `is a beautiful city .`

`[ Barcelona ]`$_{\text{ORG}}$ `won the game .`

Assume we use feature templates 1–5 defined above. We need to set some weights such that the training set is perfectly classified, and such that the norm of the weight vector is not too large (in this exercise we will make training error to be zero, and try to set a few non-zero weights). A possible weight vector could be:

$$w_{1,\text{LOC},\texttt{Barcelona}} = 1$$
$$w_{5,\text{ORG},\texttt{won}} = 2$$

Thus, with these weights, we have "learned" that `Barcelona` is a location, but that words surrounded by `won` should have a higher positive score for being organizations. Note that this knowledge is acquired via learning: the feature templates do not have prior knowledge about what words are locations or surround organizations.

Assume the following training set:

`[ Maria ]`$_{\text{PER}}$ `is beautiful .`

`[ Barcelona ]`$_{\text{LOC}}$ `is beautiful .`

`[ Jack London ]`$_{\text{PER}}$ `is nice .`

`[ Milan ]`$_{\text{LOC}}$ `is nice .`

`[ Jack Smith ]`$_{\text{PER}}$ `lives in [ Great Yarmouth ]`$_{\text{LOC}}$ `.`

`[ Maria Domenech ]`$_{\text{PER}}$ `works in [ Barcelona ]`$_{\text{LOC}}$ `.`

`[ Maria Muschatta ]`$_{\text{PER}}$ `attends school in [ Santa Maria Della Mole ]`$_{\text{LOC}}$ `.`

`[ Barcelona ]`$_{\text{ORG}}$ `won [ Milan ]`$_{\text{ORG}}$ `.`

Question: Using feature templates 1–5, set a parameter vector with as few features as possible that correctly classifies all examples in the training set.

Features for n-gram-factored sequence annotation models

Recall the factored linear models for sequence prediction, and think of a named entity task. A bigram-factored sequence model computes:

$$\begin{aligned} tags(x_{1:n}) &= \operatorname*{argmax}_{y_{1:n} \in \mathcal{Y}^n} \mathbf{w} \cdot \mathbf{f}(x, y_{1:n}) \\ &= \operatorname*{argmax}_{y_{1:n} \in \mathcal{Y}^n} \sum_{i=1}^{n} \mathbf{w} \cdot \mathbf{f}(x, i, y_{i-1}, y_i) \end{aligned} \quad (1)$$

where $x_{1:n}$ is an input sentence of n tokens (x_i is the i-th token), $y_{1:n}$ is an output sequence of n tags ($\mathcal{Y}$ is the set of valid tags). $\mathbf{f}(x, i, y_{i-1}, \mathbf{y}_i)$ is a function returning a feature vector of the bigram y_{i-1}, y_i at position i of the sentence (assume that y_0 is a special tag START *that indicates the start of the sequence). $\mathbf{w}$ is a vector of parameters of the same dimensionality of the feature vectors.*

Exercise 5.

We specify features using templates. For example, the following template captures the current word and the current tag:

$$\mathbf{f}_{1,l,a}(x, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } x_i = a \text{ and } y_i = l \\ 0 & \text{otherwise} \end{cases}$$

Write feature templates that capture the following patterns. Justify your answers if necessary.

- $\mathbf{f}_{2,a}$: the current word is the first of the sentence, it is capitalized, and its tag is a
- $\mathbf{f}_{3,s,a}$: 3-letter prefix of the current word, together with the current tag
- $\mathbf{f}_{4,w,a,b}$: the current word, the current tag, and the previous tag
- $\mathbf{f}_{5,w,v,a}$: the two previous words and the current tag
- $\mathbf{f}_{6,a,b,c}$: the two previous tags and the current tag

Exercise 6.

1. Given the training example `the/DT dog/NN saw/VBD the/DT man/NN`, if we convert it to (x, y) pairs for training a trigram-factored log-linear model for PoS tagging, which of the following pairs are in the training set?
   (a) $x = (\texttt{DT}, \texttt{NN}, \texttt{the dog saw the man}, 3); y = \texttt{NN}$
   (b) $x = (\texttt{VBD}, \texttt{DT}, \texttt{the dog saw the man}, 3); y = \texttt{VBD}$
   (c) $x = (\texttt{DT}, \texttt{NN}, \texttt{the dog saw the man}, 3); y = \texttt{VBD}$
   (d) $x = (\texttt{DT}, \texttt{NN}, \texttt{the dog saw the man}, 4); y = \texttt{NN}$
2. List all (x, y) pairs that can be generated from this training set. Assume phantom tags $y_{-1} = y_0 =$ `START`.

Exercise 7.

We want to approach a PoS tagging task with a bigram-factored log-linear model that will compute the tag for each word as:

$$tag(x_{1:n}, i) = \operatorname*{argmax}_{y_i \in \mathcal{Y}} w \cdot f(x_{1:n}, i, y_{i-1}, y_i)$$

We have defined the following feature function types:

- Type 1: Current tag is a:

$$f_{1,a}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } y_i = a \\ 0 & otherwise \end{cases}$$

- Type 2: Current word is capitalized and current tag is a:

$$f_{2,a}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } x_i \text{ is capitalized and } y_i = a \\ 0 & \text{otherwise} \end{cases}$$

- Type 3: Current tag is a and previous tag is b:

$$f_{3,a,b}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } y_{i-1} = a \text{ and } y_i = b \\ 0 & \text{otherwise} \end{cases}$$

1. Propose values for appropriate features in vector w that will correctly classify all words in the following sentences. Try to set the minimum number of non-zero weights. Proof or justification of the chosen values is required.

x:	John	programs	bugs
y:	E	V	N

x:	Mary	runs	programs
y:	E	V	N

x:	Mary	bugs	John
y:	E	V	E

x:	programs	print	results
y:	N	V	N

Exercise 8.

We are performing PoS tagging with a trigram-factored CRF, using tagset $\mathcal{T} = \{\texttt{DT}, \texttt{V}, \texttt{NN}, \texttt{ADV}, \texttt{PREP}\}$, and we defined a history as $h = \langle t_{i-2}, t_{i-1}, w_{[1:n]}, i \rangle$.

1. How many possible histories are there for a given input sequence $\mathcal{X}$ and a fixed value of i?
2. Which of the following are valid features?

$$\mathbf{f}_1(h,t) = \begin{cases} 1 & \text{if } t = \texttt{V} \text{ and } t_{i-1} = \texttt{PREP} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_2(h,t) = \begin{cases} 1 & \text{if } t = \texttt{V} \text{ and } w_{i-2} = \texttt{dog} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_3(h,t) = \begin{cases} 1 & \text{if } t = \texttt{V} \text{ and } t_{i-3} = \texttt{NN} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_4(h,t) = \begin{cases} 1 & \text{if } t = \texttt{V} \text{ and } t_{i+1} = \texttt{PREP} \text{ and } w_2 = \texttt{cow} \\ 0 & \text{otherwise} \end{cases}$$

3. Compute the global feature vector $\mathbf{f}(\mathcal{X}, \mathcal{Y})$ for the input sequence is $\mathcal{X}$ =`the dog walked to a park` and the tag sequence $\mathcal{Y}$ = `DT NN V PREP DT NN`, when using the following features:

$$\mathbf{f}_1(h,t) = \begin{cases} 1 & \text{if } t = \texttt{NN} \text{ and } w_i = \texttt{dog} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_2(h,t) = \begin{cases} 1 & \text{if } t = \texttt{NN} \text{ and } t_{i-1} = \texttt{DT} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_3(h,t) = \begin{cases} 1 & \text{if } t = \texttt{NN} \text{ and } t_{i-1} = \texttt{DT} \text{ and } w_{i-1} = \texttt{the} \\ 0 & \text{otherwise} \end{cases}$$

4. Given the history $h = (t_{i-2}, t_{i-1}, w_{[1:n]}, 5) = (\texttt{V}, \texttt{DT}, \texttt{the man saw the dog in the park}, 5)$, which of the following features yield $\mathbf{f}(h, \texttt{NN}) = 1$?

$$\mathbf{f}_1(h,t) = \begin{cases} 1 & \text{if } t = \texttt{NN} \text{ and } w_i = \texttt{dog} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_2(h,t) = \begin{cases} 1 & \text{if } t = \texttt{DT} \text{ and } w_i = \texttt{dog} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_3(h,t) = \begin{cases} 1 & \text{if } t = \texttt{NN} \text{ and } w_{i+1} = \texttt{dog} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_4(h,t) = \begin{cases} 1 & \text{if } t = \texttt{NN} \text{ and } t_{i-1} = \texttt{DT} \\ 0 & \text{otherwise} \end{cases}$$

Exercise 9.

We want to address a Named Entity Recognition task consisting in identifying diseases in medical texts. For this, we want to train a sequence classifier such as a CRF using bigram factorization (i.e. only previous and current tag hypothesis are considered). Thus, the used context is $h = (t_{i-1}, w_{[1:n]}, pos_{[1:n]}, i)$.

We use the following feature templates:

$$\mathbf{f}_1(h,t) = \begin{cases} 1 & \text{if } pos_{i-1} = \text{N} \text{ and } t_{i-1} = \text{O} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_2(h,t) = \begin{cases} 1 & \text{if } suf(w_{i-1}) = \text{'ing'} \text{ and } t_i = \text{B} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_{3,a,b}(h,t) = \begin{cases} 1 & \text{if } w_{i-1} = a \text{ and } t = b \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_{4,a,c}(h,t) = \begin{cases} 1 & \text{if } w_{i-1} = a \text{ and } pos_i = c \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_{5,a}(h,t) = \begin{cases} 1 & \text{if } w_i = a \text{ and } capitalized(w_{i-1}) \\ 0 & \text{otherwise} \end{cases}$$

Given the above templates, and the training sentence:

i	1	2	3	4	5	6	7	8	9	10	11	12	13
w	Fragile-X	syndrome	is	an	inherited	form	of	mental	retardation	involving	mitral	valve	prolapse
pos	N	N	V	D	JJ	N	P	JJ	N	V	JJ	N	N
t	B	I	O	O	O	O	O	O	O	O	B	I	I

List which feature instances would be generated for words:

a) $i = 2$ (*syndrome*)

b) $i = 10$ (*involving*)

c) $i = 11$ (*mitral*)

Exercise 10.

We are performing PoS tagging for a recently discovered alien language, using a trigram-factored CRF, using tagset $\mathcal{T} = \{\texttt{D}, \texttt{V}, \texttt{N}, \texttt{A}, \texttt{P}\}$, and we defined a history as $h = \langle t_{i-2}, t_{i-1}, w_{[1:n]}, i \rangle$.

1. How many possible histories are there for a given input sequence $\mathcal{X}$ and a fixed value of i? Justify your answer.

2. Which of the following are valid features and which are not? Justify your answer.

$$\mathbf{f}_1(h,t) = \begin{cases} 1 & \text{if } t = \texttt{V} \text{ and } t_{i-1} = \texttt{N} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_2(h,t) = \begin{cases} 1 & \text{if } t = \texttt{K} \text{ and } w_{i-2} = \texttt{skjkeg} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_3(h,t) = \begin{cases} 1 & \text{if } t = \texttt{N} \text{ and } t_{i-3} = \texttt{P} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_4(h,t) = \begin{cases} 1 & \text{if } t = \texttt{V} \text{ and } t_{i+1} = \texttt{A} \text{ and } w_2 = \texttt{wuakla} \\ 0 & \text{otherwise} \end{cases}$$

3. Compute the feature vectors $\mathbf{f}(h,t)$ for each position i, and the global feature vector $\mathbf{f}(\mathcal{X},\mathcal{Y})$ for the input sequence $\mathcal{X} =$ `grufp umdk wuakla du blha skjkeg` and the tag sequence $\mathcal{Y} =$ `P V N D N A`, when using the following features:

$$\mathbf{f}_1(h,t) = \begin{cases} 1 & \text{if } t = \texttt{N} \text{ and } w_i = \texttt{wuakla} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_2(h,t) = \begin{cases} 1 & \text{if } t = \texttt{N} \text{ and } t_{i-1} \neq \texttt{A} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{f}_3(h,t) = \begin{cases} 1 & \text{if } t = \texttt{N} \text{ and } t_{i-1} = \texttt{V} \text{ and } w_{i-1} = \texttt{umdk} \\ 0 & \text{otherwise} \end{cases}$$

Exercise 11.

Assume we have defined the following feature templates for a bigram-factored model:

- f_1: the current tag is a and the current word is capitalized

$$\mathbf{f}_{1,a}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } \text{is\_capitalized}(x_i) \text{ and } y_i = a \\ 0 & \text{otherwise} \end{cases}$$

- f_2: the current tag is a and the current word is *not* capitalized

$$\mathbf{f}_{2,a}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if not } \text{is\_capitalized}(x_i) \text{ and } y_i = a \\ 0 & \text{otherwise} \end{cases}$$

- f_3: the previous tag is a, the current tag is b and the current word is capitalized

$$\mathbf{f}_{3,a,b}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } \text{is\_capitalized}(x_i) \text{ and } y_{i-1} = a \text{ and } y_i = b \\ 0 & \text{otherwise} \end{cases}$$

- f_4: the current tag is a and the current word is w

$$\mathbf{f}_{4,a,w}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } x_i = w \text{ and } y_i = a \\ 0 & \text{otherwise} \end{cases}$$

- f_5: the current tag is a and the next word is w

$$\mathbf{f}_{5,a,w}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } x_{i+1} = w \text{ and } y_i = a \\ 0 & \text{otherwise} \end{cases}$$

- f_6: the current tag is a and the previous word is w

$$\mathbf{f}_{6,a,w}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } x_{i+1} = w \text{ and } y_i = a \\ 0 & \text{otherwise} \end{cases}$$

- f_7: the previous tag is a and the current tag is b

$$\mathbf{f}_{7,a,b}(x_{1:n}, i, y_{i-1}, y_i) = \begin{cases} 1 & \text{if } y_{i-1} = a \text{ and } y_i = b \\ 0 & \text{otherwise} \end{cases}$$

1. **Perceptron updates**

   Consider the following training example:

y	PER	PER	-	-	LOC	LOC
x	Jack	London	went	to	South	Paris

   Say we are running Perceptron, and under our current **w** the prediction given by Equation **??** is the following sequence **z**:

z	PER	LOC	-	-	-	LOC
x	Jack	London	went	to	South	Paris

   Write the perceptron update. That is, write a weight vector **g** corresponding to $\mathbf{g} = \mathbf{f}(x, y) - \mathbf{f}(x, \mathbf{z})$. Write the non-zero values only.

2. **Setting weights**

   Using the feature definitions above, write a weight vector that correctly classifies all the examples below, i.e. the weight vector should predict the correct tag sequences for all the examples. Try to set as few non-zero weights as possible. Justify your answer.

PER	-	-
Maria	is	beautiful
LOC	-	-
Barcelona	is	beautiful
PER	-	-	LOC
Jack	went	to	London
LOC	-	-
Paris	is	nice
PER	PER	-	-	LOC	LOC
Jack	London	went	to	South	Paris
ORG	-	-	ORG
Barcelona	played	against	Paris

Exercise 12.

Recall the factored linear models of the previous exercises:

$$f(\mathbf{x}_{1:n}) = \operatorname*{argmax}_{\mathbf{y}_{1:n} \in \mathcal{Y}^n} \sum_{i=1}^{n} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, i, \mathbf{y}_{i-1}, \mathbf{y}_i)$$

In order to compute $f(\mathbf{x}_{1:n})$ we can use the Viterbi algorithm as follows:

- *Define $\delta_i(a)$ to be the score of optimal sequence for $\mathbf{x}_{1:i}$ ending with $a \in \mathcal{Y}$:*

$$\delta_i(a) = \max_{\mathbf{y}_{1:i} \in \mathcal{Y}^i : \mathbf{y}_i = a} \sum_{j=1}^{i} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, j, \mathbf{y}_{j-1}, \mathbf{y}_j)$$

- *Use the following recursions, for all $a \in \mathcal{Y}$:*

$$\begin{aligned}
\delta_1(a) &= \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, 1, \text{START}, a) \quad \textit{(initialization, assuming } y_0 = \text{START}\textit{)} \\
\delta_i(a) &= \max_{b \in \mathcal{Y}} \delta_{i-1}(b) + \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, i, b, a) \quad \textit{(recursion, } \forall i > 1\textit{)} \\
\gamma_i(a) &= \operatorname*{argmax}_{b \in \mathcal{Y}} \delta_{i-1}(b) + \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, i, b, a) \quad \textit{(backtrace, } \forall i > 1\textit{)}
\end{aligned}$$

- *The optimal score for $\mathbf{x}$ is $\max_{a \in \mathcal{Y}} \delta_n(a)$*
- *The optimal sequence $\hat{\mathbf{y}}$ can be recovered through the backpointers γ*

1. **Viterbi for Trigram Models**

   Assume our model is now factored on trigrams

$$f(\mathbf{x}_{1:n}) = \operatorname*{argmax}_{\mathbf{y}_{1:n} \in \mathcal{Y}^n} \sum_{i=1}^{n} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, i, \mathbf{y}_{i-2}, \mathbf{y}_{i-1}, \mathbf{y}_i)$$

   That is, $\mathbf{f}(\mathbf{x}, i, \mathbf{y}_{i-2}, \mathbf{y}_{i-1}, \mathbf{y}_i)$ now returns a feature vector for a trigram of the output sequence *ending* at position i. You can assume that $\mathbf{y}_0$ and $\mathbf{y}_{-1}$ are special tags START. Redefine the δ and γ variables and the recursions to compute them.

2. **Viterbi for Trigram Models, Variation**

   Assume our model is now factored on trigrams, but with a slight change in the definition:

$$f(\mathbf{x}_{1:n}) = \operatorname*{argmax}_{\mathbf{y}_{1:n} \in \mathcal{Y}^n} \sum_{i=1}^{n} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, i, \mathbf{y}_{i-1}, \mathbf{y}_i, \mathbf{y}_{i+1})$$

   Now $\mathbf{f}(\mathbf{x}, i, \mathbf{y}_{i-1}, \mathbf{y}_i, \mathbf{y}_{i+1})$ returns a feature vector for a trigram of the output sequence *centered* at position i. You can assume that $\mathbf{y}_0$ is a special tag START and that $\mathbf{y}_{n+1}$ is a special END tag. Redefine the δ and γ variables and the recursions to compute them.