NLP Exercise 1

Eyal Orbach - ID 015369317
Daniel Juravski - ID 206082323

1. Describe how you handled unknown words in hmm1:
We used 2 main ways:
a) On train, each word and tag that appears only once (together), was counted as 'unk' word and added with the above tag.
On test, when there is a word that doesn't appear in e.mle, that word is treated as if it were the 'unk' word.

b) Signature:
- *) looking for number patterns:
  - floats
  - number-number
  - number:number
- *) looking for 'ing' suffix
- *) looking for 'ed' suffix
- *) looking for capital letter prefix

2. Describe your pruning strategy in the viterbi hmm:
We used pruning by two different mechanisms
a) Prune by emission probability:
-When examining a word, in test time, that we have encountered in train time, we will only consider tags that have appeared for that word.
-When examining a word that we have not yet encountered all tags will be evaluated with some probability based on the 'unk' word. Probability for 'unk' is based on the train data as described above, with at least some minimal probability for every tag.

b) Prune by transition probability:
-We have eliminated some combinations that are either illegitimate (i.e. `NNP start`) or do not make any sense (i.e. `: POS`)
To deal with the possibility that such sequences will appear in a legitimate use case we did not think of, this pruning will be dropped if it prevents the algorithm from reaching a possible sequence.

3. Report your test scores:

Data \ Tagger	hmm-greedy	hmm-viterbi	memm-greedy	memm-viterbi
POS Data	94.02%	95.91%	94.87%	95.70%
NER Data	94.75%	96.09%	93.71%	95.83%

NER Full Results:

```
==============================================================================
HMM-greedy:
-----------
Accuracy: 0.947516382954

All-types       Prec:0.735274318411 Rec:0.775195173882 F1:0.754707203317
       LOC      Prec:0.810560696788 Rec:0.817682591982 F1:0.81410606889
      MISC      Prec:0.712581344902 Rec:0.763953488372 F1:0.737373737374
       PER      Prec:0.728013029316 Rec:0.874755381605 F1:0.794666666667
       ORG      Prec:0.657718120805 Rec:0.620253164557 F1:0.638436482085

==============================================================================
HMM-viterbi:
------------
Accuracy: 0.960952344023

All-types       Prec:0.792325816223 Rec:0.820637964093 F1:0.806233410395
       LOC      Prec:0.822536744692 Rec:0.882078225336 F1:0.851267605634
      MISC      Prec:0.759219088937 Rec:0.832342449465 F1:0.794100964265
       PER      Prec:0.815418023887 Rec:0.859267734554 F1:0.836768802228
       ORG      Prec:0.741983594333 Rec:0.693379790941 F1:0.716858789625

==============================================================================
MEMM-greedy:
------------
Accuracy: 0.937182519679

All-types       Prec:0.639515314709 Rec:0.841638981174 F1:0.726785885053
       LOC      Prec:0.677735438214 Rec:0.908096280088 F1:0.776184538653
      MISC      Prec:0.528199566161 Rec:0.863475177305 F1:0.655450874832
       PER      Prec:0.734527687296 Rec:0.81212484994 F1:0.771379703535
       ORG      Prec:0.533184190902 Rec:0.78227571116 F1:0.634146341463

==============================================================================
MEMM-viterbi:
-------------
Accuracy: 0.95835433712

All-types       Prec:0.763884214069 Rec:0.84149054505 F1:0.800811573747
       LOC      Prec:0.82362547632 Rec:0.855769230769 F1:0.839389736477
      MISC      Prec:0.716919739696 Rec:0.837769328264 F1:0.772647574518
       PER      Prec:0.843105320304 Rec:0.889461626575 F1:0.865663322185
       ORG      Prec:0.605518269948 Rec:0.744271310724 F1:0.667763157895
==============================================================================
```

Attempts to improve the MEMM tagger for the NER data:

Attempt 1, For each word we ran over the lexicons files,
if the word appeared in a lexicon file, then we added a feature '`lexicon=<name of file it appeared in>`'
This produced a very slight improvement:

```
Accuracy: 0.958722711233

All-types       Prec:0.796334361083 Rec:0.782396499495 F1:0.789303904924
       LOC      Prec:0.81332628239 Rec:0.837234621666 F1:0.825107296137
      MISC      Prec:0.81 Rec:0.702819956616 F1:0.752613240418
       PER      Prec:0.825951016154 Rec:0.860477741585 F1:0.842860941239
       ORG      Prec:0.714983713355 Rec:0.654735272185 F1:0.683534449202
```

Attempt 2, For each word we ran over the lexicons files,
if the word appeared in a lexicon file then we also checked if it appears in the exact context, meaning for the word museum, if when searching the lexicons we ran into "museum of modern art" than we'd check that the following words in the sentence are "of modern art", only if this condition was met then we add the feature '`lexicon=<name of file it appeared in>`'
This produced more improvement:

```
Accuracy: 0.959478847571

All-types       Prec:0.821873886712 Rec:0.77650622686 F1:0.798546209761
       LOC      Prec:0.851134477034 Rec:0.837234621666 F1:0.844127332602
      MISC      Prec:0.8325 Rec:0.722342733189 F1:0.773519163763
       PER      Prec:0.856906989543 Rec:0.845276872964 F1:0.851052200055
       ORG      Prec:0.716806722689 Rec:0.636092468307 F1:0.67404188068
```

4. Is there a difference in behavior between the hmm and maxent taggers? discuss.
The MEMM tagger is more generic opposed to the HMM Taggers which used specific domain knowledge such as common word suffixes like 'ly', 'ing', date detection and so on.
The learning time for the MEMM taggers was obviously longer.

5. Is there a difference in behavior between the datasets? discuss.
The difference are:

1. NER tag set size is 10
   POS tag set size is ~45
   If the tag set is smaller, then there is more probability to predict the correct tag.
2. The most common tag of the NER tag set is 'O'. The more the model will predict it, the precision will likely improve.
3. In the NER data set, the sentences are much shorter, what makes the viterbi algorithms (and surely the greedy) to run much faster.

6. What will you change in the hmm tagger to improve accuracy on the named entities data?
To improve the HMM tagger on the 'named entities' data we will need to implement specific domain knowledge that is relevant to named entities, using either the lexicons or generic rules like recognizing capital letters usage.

7. What will you change in the memm tagger to improve accuracy on the named entities data, on top of what you already did?
On top of the usage of lexicons we added we could also predict POS on the given text and use that information to further predict when a named entity could appear giving the POS tags on previous words.

8. Why are span scores lower than accuracy scores?
The span-based scores are lower than the accuracy scores because when we compute the accuracy scores, we compare each pred word tag to the gold word tag.

At this case (F scores) we compare all the span as a one word. We are looking for all the tags in the span words to be correct. That way we need to be correct about every word in the span to be correct on the whole span unit.