Project 1: developing an NE tagger

- Download the training, development, and test set from Latte (proj1-data.tgz, to be uploaded)
- Also download the evaluation program (evaluate-head.py)
- Use the recommended Mallet CRF tagger, or CRF++, or any machine learning classifier of your choice
  - Choose your classifier wisely, because accuracy matters to your grade
- Most of the work involving extracting features and training a sequential learner. Using the development set to tune your features.
- Use the test set and the evaluation program to report the accuracy of your tagger
- .

Mallet sequential learner

- Mallet 2.0.7 accessible via CS computers
  - /home/j/clp/chinese/tools/mallet-2.0.7
- Using Mallet:
  - Setting up a shell script (let's assume it's called 'mallet-tag')

```
#!/bin/sh
MALLET_HOME=/home/j/clp/chinese/tools/mallet-2.0.7
Export CLASSPATH=$MALLET_HOME/class:$MALLET_HOME/mallet-deps.jar
Java –mx1000m cc.mallet.fst.SimpleTagger "$@"
```

  - Training a tagger with Mallet:

```
mallet-tag –train true –model-file <MODELFILE> <TRAINING-DATA>
```

  - Using the tagger:

```
mallet-tag –include-input –model-file <MODELFILE> <INPUT>   > output.txt
```

Data

- Training, dev, and test sets
  - Training for training models (train.gold)
  - Dev for feature development (dev.gold, dev.raw)
  - Test for final evaluation (test.gold, test.raw)
- Data split: train/dev/test = 80/20/28, 128 total files
- Ideally data should be provided in individual files, but logistically this is difficult.

Sample feature vectors (training)

Syrian Capitalized nextword=President next_capitalized O
President Capitalized prevword=Syrian prev_capitalized nextword=Travels next_capitalized O
Travels Capitalized prevword=President prev_capitalized nextword=To next_capitalized O
To Capitalized prevword=Travels prev_capitalized nextword=Egypt next_capitalized O
Egypt Capitalized prevword=To prev_capitalized O

Bashar Capitalized prevword=, nextword=Assad next_capitalized B-PER
Assad Capitalized prevword=Bashar prev_capitalized nextword=met I-PER
met prevword=Assad prev_capitalized nextword=Sunday next_capitalized O
Sunday Capitalized prevword=met nextword=with O
with prevword=Sunday prev_capitalized nextword=Egyptian next_capitalized O
Egyptian Capitalized prevword=with nextword=President next_capitalized O
President Capitalized prevword=Egyptian prev_capitalized nextword=Hosni next_capitalized B-PER
Hosni Capitalized prevword=President prev_capitalized nextword=Mubarak next_capitalized B-PER
Mubarak Capitalized prevword=Hosni prev_capitalized nextword=in I-PER
in prevword=Mubarak prev_capitalized nextword=talks O
talks prevword=in nextword=on O
on prevword=talks nextword=Mideast next_capitalized O
Mideast Capitalized prevword=on nextword=peace O
peace prevword=Mideast prev_capitalized nextword=and O

Output is a model

Feature vectors (training)

Syrian Capitalized nextword=President next_capitalized O
President Capitalized prevword=Syrian prev_capitalized nextword=Travels next_capitalized O
Travels Capitalized prevword=President prev_capitalized nextword=To next_capitalized O
To Capitalized prevword=Travels prev_capitalized nextword=Egypt next_capitalized O
Egypt Capitalized prevword=To prev_capitalized O

Bashar Capitalized prevword=, nextword=Assad next_capitalized B-PER
Assad Capitalized prevword=Bashar prev_capitalized nextword=met I-PER
met prevword=Assad prev_capitalized nextword=Sunday next_capitalized O
Sunday Capitalized prevword=met nextword=with O
with prevword=Sunday prev_capitalized nextword=Egyptian next_capitalized O
Egyptian Capitalized prevword=with nextword=President next_capitalized O
President Capitalized prevword=Egyptian prev_capitalized nextword=Hosni next_capitalized B-PER
Hosni Capitalized prevword=President prev_capitalized nextword=Mubarak next_capitalized B-PER
Mubarak Capitalized prevword=Hosni prev_capitalized nextword=in I-PER
in prevword=Mubarak prev_capitalized nextword=talks O
talks prevword=in nextword=on O
on prevword=talks nextword=Mideast next_capitalized O
Mideast Capitalized prevword=on nextword=peace O
peace prevword=Mideast prev_capitalized nextword=and O

Sparse data format

Output is a model

Feature vectors (testing)

U.S. nextword=District next_capitalized
District Capitalized prevword=U.S. nextword=Court next_capitalized
Court Capitalized prevword=District prev_capitalized nextword=Judge next_capitalized
Judge Capitalized prevword=Court prev_capitalized nextword=Murray next_capitalized
Murray Capitalized prevword=Judge prev_capitalized nextword=Schwartz next_capitalized
Schwartz Capitalized prevword=Murray prev_capitalized nextword=in
in prevword=Schwartz prev_capitalized nextword=Wilmington next_capitalized
Wilmington Capitalized prevword=in nextword=,
, prevword=Wilmington prev_capitalized nextword=Del.
Del. prevword=, nextword=,
, prevword=Del. nextword=ruled
ruled prevword=, nextword=that
that prevword=ruled nextword=Camelot next_capitalized
Camelot Capitalized prevword=that nextword=Music next_capitalized
Music Capitalized prevword=Camelot prev_capitalized nextword=could
could prevword=Music prev_capitalized nextword=not
not prevword=could nextword=deduct
deduct prevword=not nextword=interest
interest prevword=deduct nextword=on
on prevword=interest nextword=loans
loans prevword=on nextword=it

Sparse data format

Output is a list of NE tags

CRF++

- http://crfpp.googlecode.com/svn/trunk/doc/index.html#format
- Uses a feature template
- Dense feature format

He	PRP	B-NP
reckons		B-VP
the		B-NP
current	JJ	I-NP
account	NN	I-NP

He	PRP	B-NP
reckons	VBZ	B-VP
the	DT	B-NP
current	JJ	I-NP
account	NN	I-NP

CRF++ feature template

```
# Unigram
U00:%x[-2,0]
U01:%x[-1,0]
U02:%x[0,0]
U03:%x[1,0]
U04:%x[2,0]
U05:%x[-1,0]/%x[0,0]
U06:%x[0,0]/%x[1,0]

U10:%x[-2,1]
U11:%x[-1,1]
U12:%x[0,1]q
U13:%x[1,1]
U14:%x[2,1]
U15:%x[-2,1]/%x[-1,1]
U16:%x[-1,1]/%x[0,1]
U17:%x[0,1]/%x[1,1]
U18:%x[1,1]/%x[2,1]

U20:%x[-2,1]/%x[-1,1]/%x[0,1]
U21:%x[-1,1]/%x[0,1]/%x[1,1]
U22:%x[0,1]/%x[1,1]/%x[2,1]

# Bigram
B
```

```
% crf_learn template_file train_file model_file

% crf_test -m model_file test_files
```

What to turn in

- **Project due on 2/5. Submit your feature extraction code as well as the output of your tagger on the test set to Latte**
- **Each team is also asked to make a 15-min presentation reporting the classifier you used, your features, and your results.**
  - Include in your report the contribution of each feature type
  - Include a baseline from features mined from the literature, and improvement from features you devise on your own over the baseline
- **You project will be evaluated on the accuracy of your tagger, and the creativity of your features.**
  - Higher accuracy corresponds to higher grades
  - Bonus points for novel features not found in the literature