Final review

LING572

Fei Xia

Week 10: 03/11/2010

Topics covered

- Supervised learning: seven algorithms
  - kNN, NB: training and decoding
  - DT, TBL: training and decoding (with binary features)
  - MaxEnt: training (GIS) and decoding
  - SVM: decoding, tree kernel
  - CRF**

- Unsupervised learning: EM**
- Semi-supervised learning: self-training, co-training**

Other topics

- From LING570:
  - Introduction to classification task
  - Mallet
  - Beam search

- Information theory: entropy, KL divergence, info gain
- Feature selection: e.g., chi-square, feature frequency
- Multi-class to binary conversion: e.g., one-vs-all, all-pairs

- Sequence labeling problem
- Reranking

Assignments

- Hw1: Probability, Info theory, Mallet
- Hw2: Decision tree
- Hw3: Naïve Bayes
- Hw4: kNN and feature selection
- Hw5: MaxEnt decoder
- Hw6: Beam search
- Hw7: one-vs-all, all-pairs
- Hw8: SVM decoder
- Hw9: TBL trainer and decoder

Main steps for solving a classification task

- Formulate the problem
- Define features
- Prepare training and test data
- Select ML learners
- Implement the learner
- Run the learner
  - Tune parameters on the dev data
  - Error analysis
  - Conclusion

Learning algorithms

Generative vs. discriminative models

- Joint (generative) models estimate P(x,y) by maximizing the likelihood: $P(X,Y|\theta)$
  - Ex: n-gram models, HMM, Naïve Bayes, PCFG
  - Training is trivial: just use relative frequencies.

- Conditional (discriminative) models estimate P(y|x) by maximizing the conditional likelihood: $P(Y|X, \theta)$
  - Ex: MaxEnt, SVM, CRF, etc.
  - Training is harder.

Parametric vs. non-parametric models

- Parametric model:
  - The number of parameters do not change w.r.t. the number of training instances
  - Ex: NB, MaxEnt, linear SVM

- Non-parametric model:
  - More examples could potentially mean more complex classifiers.
  - Ex: kNN, non-linear SVM, DT, TBL

Feature-based vs. kernel-based

- Feature-based:
  - Representing x as a feature vector
  - Need to define features
  - Ex: DT, NB, MaxEnt, TBL, CRF, …

- Kernel-based:
  - Calculating similarity between two objects
  - Need to define similarity/kernel function
  - Ex: kNN, SVM

DT and TBL

- DT:
  - Training: build the tree
  - Testing: traverse the tree

- TBL:
  - Training: create a transformation list
  - Testing: go through the list

- Both use the greedy approach
  - DT chooses the split that maximizes info gain, etc.
  - TBL chooses the transformation that reduces the errors the most.

NB and MaxEnt

- NB:
  - Training: estimate $P(c)$ and $P(f|c)$
  - Testing: calculate $P(y)P(x|y)$

- MaxEnt:
  - Training: estimate the weight for each (f, c)
  - Testing: calculate $P(y|x)$

- Differences:
  - generative vs. discriminative models
  - MaxEnt does not assume features are conditionally independent

kNN and SVM

- Both work with data through “similarity” functions between vectors.

- kNN:
  - Training: Nothing
  - Testing: Find the nearest neighbors

- SVM
  - Training: Estimate the weights of training instances and b.
  - Testing: Calculating f(x), which uses all the SVs

MaxEnt and SVM

- Both are discriminative models.

- Start with an objective function and find the solution to an optimization problem by using
  - Lagrangian, the dual problem, etc.
  - Iterative approach: e.g., GIS
  - Quadratic programming

  ➔ numerical optimization

HMM, MaxEnt and CRF

- linear CRF is like HMM + MaxEnt
  - Training is similar to training for MaxEnt
  - Decoding is similar to Viterbi for HMM decoding
  - Features are similar to the ones for MaxEnt

Comparison of three learners

	Naïve Bayes	MaxEnt	SVM
Modeling	Maximize $P(X,Y \mid \lambda)$	Maximize $P(Y \mid X, \lambda)$	Maximize the margin
Training	Learn P(c) and P(f|c)	Learn λ_i for feature function	Learn α_i for each (x_i, y_i)
Decoding	Calc P(y)P(x|y)	Calc P(y|x)	Calc f(x)
Things to decide	Delta for smoothing Features	Regularization Training alg Features	Regularization Training alg Kernel function C for penalty

Questions for each method

- Modeling:
  - what is the model?
  - How does the decomposition work?
  - What kind of assumption is made?
  - How many model parameters?
  - How many "tuned" (or non-model) parameters?
  - How to handle multi-class problem?
  - How to handle non-binary features?
  - …

Questions for each method (cont)

- Training: how to estimate parameters?
- Decoding: how to find the "best" solution?
- Weaknesses and strengths?
  - parametric?
  - generative/discriminative?
  - performance?
  - robust? (e.g., handling outliners)
  - prone to overfitting?
  - scalable?
  - efficient in training time? Test time?

Implementation issues

Implementation issue

- Take the log: $logP(X_1, ..., X_n) = log\prod_i P(X_i|X_1, ..., X_{i-1})$
$= \sum_i logP(X_i|X_1, ..., X_{i-1})$

- Ignore some constants:

$$P(d_i|c) = P(|d_i|)|d_i|!\prod_{k=1}^{|V|} \frac{P(w_k|c)^{N_{ik}}}{N_{ik}!}$$

- Increase small numbers before dividing

$$P(c1|x) = \frac{P(x,c1)}{P(x)} = \frac{P(x,c1)}{P(x,c1)+P(x,c2)+\ldots}$$

$logP(x, c_1)$ is -200, $logP(x, c_2)$ is -201.

Implementation issue (cont)

- Reformulate the formulas: e.g., Naïve Bayes

$$P(x, c)$$

$$= P(c) \prod_{w_k \in d_i} P(w_k|c) \ \prod_{w_k \notin d_i} (1 - P(w_k|c))$$

$$= P(c) \prod_{w_k \in d_i} \frac{P(w_k|c)}{1 - P(w_k|c)} \ \prod_{w_k} (1 - P(w_k|c))$$

- Store the useful intermediate results

$$\prod_{w_k} (1 - P(w_k|c))$$

An example: calculating model expectation in MaxEnt

$$E_p f_j = \frac{1}{N} \sum_{i=1}^{N} \sum_{y \in Y} p(y \mid x_i) f_j(x_i, y)$$

for each instance x

  calculate P(y|x) for every $y \in Y$

  for each feature t in x

    for each $y \in Y$

      model_expect [t] [y] += 1/N * P(y|x)

Another example: Finding the best transformation (Hw9)

- Conceptually

for each possible transformation

go through the data and calculate the net gain

- In practice

go through the data and calculate the net gain of all applicable transformations

More advanced techniques

- In each iteration

  calculate net gain for each transformation

  choose the best transformation and apply it

- calculate net gain for each transformation

  in each iteration

  choose the best transformation and apply it

  update net gain for each transformation

What’s next?

What’s next (for LING572)?

- Today:
  - Course evaluation

- Hw9: Due 11:45pm today

What's next (beyond ling572)?

- Supervised learning:
  - Covered algorithms: e.g., L-BFGS for MaxEnt, training for SVM, math proof
  - Other algorithms: e.g., CRF, Graphical models
  - Other approaches: e.g., Bayesian

- Using algorithms:
  - Formulate the problem
  - Select features
  - Choose/compare ML algorithms

What's next? (cont)

- Semi-supervised learning:
  - LU: labeled data and unlabeled data
  - PU: labeled positive data and unlabeled data

- Unsupervised learning

- Using them for real applications: LING573