— INF4820 — Algorithms for AI and NLP

Summing up
Exam preparations

Stephan Oepen & Erik Velldal

Language Technology Group (LTG)

November 25, 2015

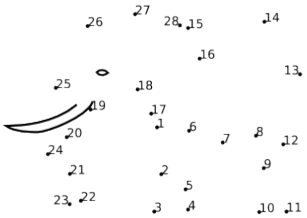
Topics for today

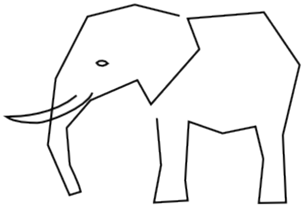
- Summing-up
- High-level overview of the most important points
- Practical details regarding the final exam
- Sample exam

Problems we have dealt with

- How to model similarity relations between pointwise observations, and how to represent and predict group membership.
- Sequences
  - Probabilities over strings: n-gram models: Linear and surface oriented.
  - Sequence classification: HMMs add one layer of abstraction; class labels as hidden variables. But still only linear.
- Grammar; adds hierarchical structure
  - Shift focus from "sequences" to "sentences".
  - Identifying underlying structure using formal rules.
    - Declarative aspect: formal grammar.
    - Procedural aspect: parsing strategy.
  - Learn probability distribution over the rules for scoring trees.

Connecting the dots...





What have we been doing?

- Data-driven learning
- by counting observations
- in context;
  - feature vectors in semantic spaces; bag-of-words, etc.
  - previous n-1 words in n-gram models
  - previous n-1 states in HMMs
  - local sub-trees in PCFGs

Data structures

- Abstract
  - Focus: How to think about or conceptualize a problem.
  - E.g. vector space models, state machines, graphical models, trees, forests, etc.
- Low-level
  - Focus: How to implement the abstract models above.
  - E.g. vector space as list of lists, array of hash-tables etc. How to represent the Viterbi trellis?

Common Lisp

- Powerful high-level language with long traditions in A.I.

Some central concepts we've talked about:

- Functions as first-class objects and higher-order functions.
- Recursion (vs iteration and mapping)
- Data structures (lists and cons cells, arrays, strings, sequences, hash-tables, etc.; effects on storage efficency vs look-up efficency)

(PS: Fine details of Lisp syntax will not be given a lot of weight in the final exam, but you might still be asked to e.g., write short functions or provide an interpretation of a given S-expression, or reflect on certain design decisions for a given programing problem.)

Vector space models

- Data representation based on a spatial metaphor.
- Objects modeled as feature vectors positioned in a coordinate system.
- Semantic spaces = VS for distributional lexical semantics
- Some issues:
  - Usage = meaning? (The distributional hypothesis)
  - How do we define context / features? (BoW, n-grams, etc)
  - Text normalization (lemmatization, stemming, etc)
  - How do we measure similarity? Distance / proximity metrics. (Euclidean distance, cosine, dot-product, etc.)
  - Length-normalization (ways to deal with frequency effects / length-bias)
  - High-dimensional sparse vectors (i.e. few active features; consequences for low-level choice of data structure, etc.)

Two categorization tasks in machine learning

Classification

- Supervised learning from labeled training data.
- Given data annotated with predefinded class labels, learn to predict membership for new/unseen objects.

Cluster analysis

- Unsupervised learning from unlabeled data.
- Automatically forming groups of similar objects.
- No predefined classes; we only specify the similarity measure.

- Some issues;
  - Measuring similarity
  - Representing classes (e.g. exemplar-based vs. centroid-based)
  - Representing class membership (hard vs. soft)

Classification

- Examples of vector space classifiers: Rocchio vs. kNN
- Some differences:
  - Centroid- vs exemplar-based class representation
  - Linear vs non-linear decision boundaries
  - Assumptions about the distribution within the class
  - Complexity in training vs complexity in prediction
- Evaluation:
  - Accuracy, precision, recall and F-score.
  - Multi-class evaluation: Micro- / macro-averaging.

Clustering

- Hierarchical vs. flat / partitional

Flat clustering

- Example: k-Means.
- Partitioning viewed as an optimization problem:
- Minimize the within-cluster sum of squares.
- Approximated by iteratively improving on some initial partition.
- Issues: initialization / seeding, non-determinism, sensitivity to outliers, termination criterion, specifying k, specifying the similarity function.

Hierarchical clustering

Agglomerative clustering

- Bottom-up hierarchical clustering
- Resulting in a set of nested partitions, often visualized as a dendrogram.
- Issues:
  - Linkage criterions — how to measure inter-cluster similarity:
    - Single, Complete, Centroid, or Average Linkage
  - Cutting the tree

Divisive clustering

- Top-down hierarchical clustering
- Generates the tree by iteratively applying a flat clustering method.

Structured Probabilistic Models

- Switching from a geometric view to a probability distribution view.
- Model the probability that elements (words, labels) are in a particular configuration.
- These models can be used for different purposes.
- We looked at many of the same concepts over structures that were

linear or **hierarchical**

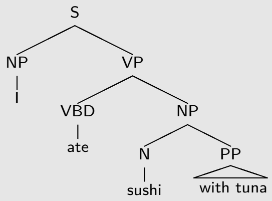
What are we Modelling?

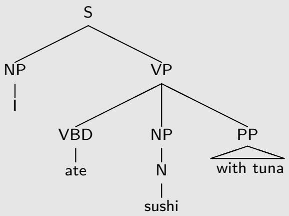
Linear

- which string is most likely:
  - *How to recognise speech* vs. *How to wreck a nice beach*
- which tag sequence is most likely for *flies like flowers*:
  - **NNS VB NNS** vs. **VBZ P NNS**

Hierarchical

- which tree structure is most likely:





The Models

Linear

- n-gram language models
  - chain rule combines conditional probabilities to model context:

$$P(w_1 \cap w_2 \cdots \cap w_{n-1} \cap w_n) = \prod_{i=1}^{n} P(w_i | w_0^{i-1})$$

  - Markov assumption allows us to limit the length of context
- Hidden Markov Model
  - added a hidden layer of abstraction: PoS tags
  - also uses the chain rule with the Markov assumption

$$P(S, O) = \prod_{i=1}^{n} P(s_i | s_{i-1}) P(o_i | s_i)$$

Hierarchical

- (Probabilistic) Context-Free Grammars (PCFGs)
  - hidden layer of abstraction: trees
  - chain rule over (P)CFG rules:

$$P(T) = \prod_{i=1}^{n} P(R_i)$$

Maximum Likelihood Estimation

Linear

- estimate n-gram probabilities:

$$P(w_n|w_1^{n-1}) \approx \frac{C(w_1^n)}{C(w_1^{n-1})}$$

- estimate HMM probabilities:

transition:

$$P(s_i|s_{i-1}) \approx \frac{C(s_{i-1}s_i)}{C(s_{i-1})}$$

emission:

$$P(o_i|s_i) \approx \frac{C(o_i : s_i)}{C(s_i)}$$

Hierarchical

- estimate PCFG rule probabilities:

$$P(\beta_1^n|\alpha) \approx \frac{C(\alpha \to \beta_1^n)}{C(\alpha)}$$

Processing

Linear

- use the n-gram models to calculate the probability of a string
- HMMs can be used to:
  - calculate the probability of a string
  - find the most likely state sequence for a particular observation sequence

Hierarchical

- A CFG can recognise strings that are a valid part of the defined language.
- A PCFG can calculate the probability of a tree (where the sentence is encoded by the leaves).

Dynamic Programming

Linear

- In an HMM, our sub-problems are prefixes to our full sequence.
- The Viterbi algorithm efficiently finds the most likely state sequence.
- The Forward algorithm efficiently calculates the probability of the observation sequence.

Hierarchical

- During (P)CFG parsing, our sub-problems are sub-trees which cover sub-spans of our input.
- Chart parsing efficiently explores the parse tree search space.
- The Viterbi algorithm efficiently finds the most likely parse tree.

Evaluation

Linear

- Tag accuracy is the most common evaluation metric for POS tagging, since usually the number of words being tagged is fixed.

Hierarchical

- Coverage is a measure of how well a CFG models the full range of the language it is designed for.
- The ParsEval metric evaluates parser accuracy by calculating the precision, recall and F_1 score over labelled constituents.

Reading List

- Both the lecture notes (slides) and the background reading specified in the lecture schedule (at the course page) are obligatory reading.
- We also expect that you have looked at the provided model solutions for the exercises.

Final Written Examination

When / where:

- 7 December at 14:30 (4 hours)
- Check StudentWeb for your assigned location.

The exam

- Just as for the lecture notes, the text will be in *English* (but you're free to answer in either English or Norwegian Bokmål/Nynorsk).
- When writing your answers, remember. . .
  - ~~Less~~ more is more! (As long as it's relevant.)
  - Aim for high recall *and* precision.
  - Don't just list keywords; spell out what you think.
  - If you see an opportunity to show off terminology, seize it.
  - Each question will have points attached (summing to 100) to give you an idea of how they will be weighted in the grading.

Finally, Some Statistics

- 61 submitted for Problem Set (1), 46 for Problem set (3b)
- 45 qualified for the final exam ...
- ... some with a larger margin than others
- Two of you stand out in terms of points throughout the term
- A total of 52–54 points (of 58), we think, is no small accomplishment
- And the 'winners' are:
  - Ole Jørgen Brønner
  - Elena Volkova
- Great work — Congratulations!

After INF4820

- Please remember to participate in the course evaluation hosted by FUI.
  - Even if this means just repeating the comments you already gave for the midterm evaluation.
  - While the midterm evaluation was only read by us, the FUI course evaluation is distributed department-wide.
- Another course of potential interest running in the spring: INF3800 - Search technology
  - Open to MSc students as INF4800.
  - Also based on the book by Manning, Raghavan, & Schütze (2008): Introduction to Information Retrieval

Sample Exam

- Please read through the complete exam once before starting to answer questions. About thirty minutes into the exam, the instructor will come around to answer any questions of clarification (including English terminology).
- As discussed in class, the exam is only given in English, but you are free to answer in any of Bokmal, English, or Nynorsk.
- To give you an idea about the relative weighting of different questions during grading, we've assigned points to each of them (summing to 100).