# *INF4820: Algorithms for Artificial Intelligence and Natural Language Processing*

## Wrap-Up and Exam Preparation

Stephan Oepen & Milen Kouylekov

Language Technology Group (LTG)

November 26, 2014

# *INF4820: Algorithms for Artificial Intelligence and Natural Language Processing*

## Wrap-Up and Exam Preparation

Stephan Oepen & Milen Kouylekov

Language Technology Group (LTG)

November 26, 2014

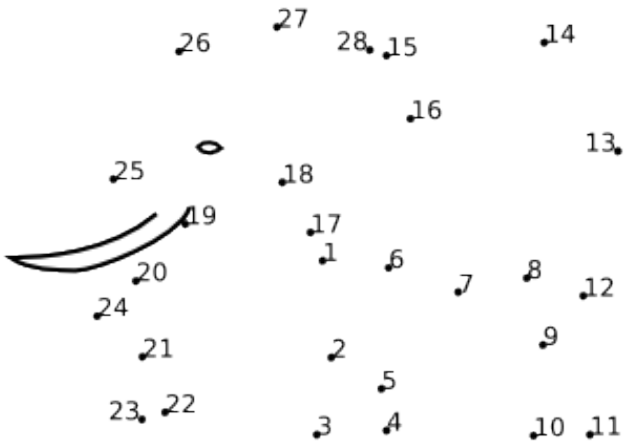
# Topics for Today

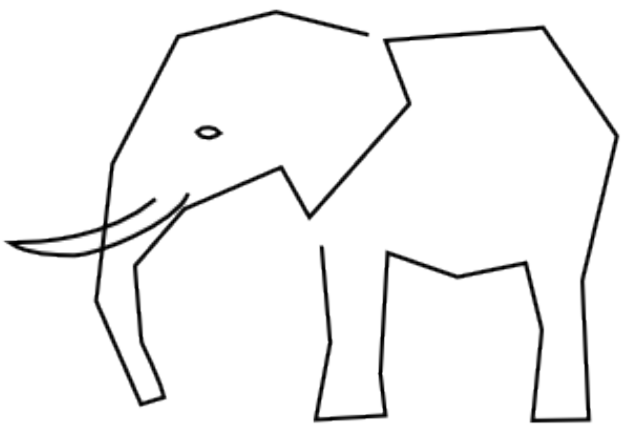
- Summing-up
- High-level overview of the most important points
- Practical details regarding the final exam

# 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;
  - Representing classes (e.g. exemplar-based vs. centroid-based)
  - Representing class membership (hard vs. soft)

# Classification

- Examples of vector space classifiers: Rocchio vs. $k$NN
- Some differences:
  - Centroid- vs exemplar-based class representation
  - Linear vs non-linear decision boundaries
  - Complexity in training vs complexity in prediction
  - Assumptions about the distribution within the class
- 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 vector space 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**

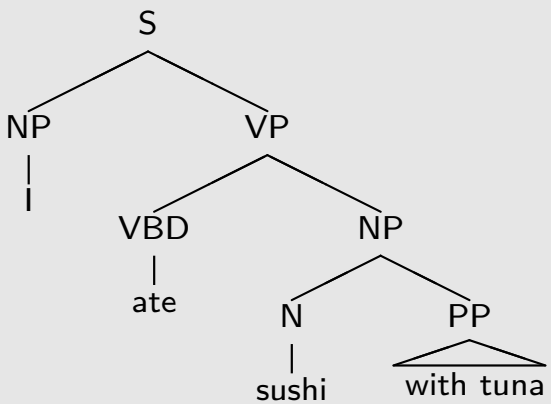
# What are We Modelling?

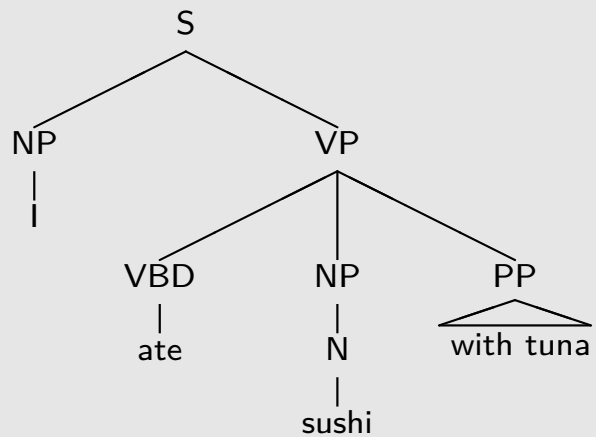
## Linear

- 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

- (Probabilitic) 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:

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

# 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:

- 5 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 (and Prizes)

- 49 submitted for Problem Set (1), 39 for Problem set (3b)
- 35 qualified for the final exam ...
- ... some with a larger margin than others
- Three of you tied for the maximum sum of points
- A total of 54 points (of 74), we think, is an accomplishment
- And the winners are:
  - Johanne Håøy Horn
  - Ole Kristian Stumpf
  - Siver Kjelberg Volle
- 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's 2008; Introduction to Information Retrieval

# Sample Exam 2014

- 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).

# (1) Common Lisp

- When working with vector space representations, we are often dealing with vectors that are simultaneously both sparse (i.e., they have a low ratio of non-zero elements) and extremely high-dimensional. Discuss some relevant choices for data structure when implementing such vectors in Lisp (e.g., based on lists, arrays, or hash-tables), highlighting the advantages and disadvantages of different approaches (in terms of storage efficiency, look-up performance, etc). *[8 points]*

- Write two versions of a function swap; one based on recursion and one based on iteration. The function should take three parameters – x, y and list – where the goal is to replace every element matching x with y in the list list. This is an example of the expected behavior:

  *(swap "foo" "bar" '("zap" "foo" "foo" "zap" "foo"))*
  *-> ("zap" "bar" "bar" "zap" "bar")*

  Try to avoid using destructive operations if you can. *[7 points]*

# (2) Classification & Clustering

- A common "pre-processing" step when working with vector space models is to apply length normalization to the vectors. Explain what is meant by this and the various effects it can have. *[10 points]*
- Describe the method of kNN classification. Briefly discuss its advantages and disadvantages. *[10 points]*
- Describe the method of k-Means clustering. Briefly discuss its advantages and disadvantages. *[10 points]*

# (3) Linear Structures

- How exactly does an n-gram model compute the probability P (s) for a string $s = w_1^n$ , assuming a bigram model? State the central assumption made in this modelling approach. *[5 points]*
- In a few sentences, explain how Hidden Markov Models extends on simple n-gram models. *[4 points]*
- Give the general formula used to calculate the values in the trellis for the Forward algorithm. Using the transition and emission probabilities given in Tables 1a and 1b and an observation sequence of flies like fruit, show the calculations made by the Forward algorithm for the first two columns of the trellis. (You don't need to solve the calculation — we just want to see that you know which formulae are used, and which numbers go in to them.) *[7 points]*
- Briefly state the differences between the Forward and Viterbi algorithms, both in terms of what they calculate, and in the details of their implementations. *[5 points]*

# Transition & Emission Probability

| | | | |
|---|---|---|---|
| $P(N\|\langle S\rangle) = \frac{6}{8}$ | $P(V\|\langle S\rangle) = \frac{1}{8}$ | $P(P\|\langle S\rangle) = \frac{1}{8}$ | |
| $P(N\|N) = \frac{1}{8}$ | $P(V\|N) = \frac{2}{8}$ | $P(P\|N) = \frac{1}{8}$ | $P(\langle/S\rangle\|N) = \frac{4}{8}$ |
| $P(N\|V) = \frac{2}{3}$ | $P(V\|V) = 0$ | $P(P\|V) = \frac{1}{3}$ | $P(\langle/S\rangle\|V) = 0$ |
| $P(N\|P) = \frac{4}{4}$ | $P(V\|P) = \frac{1}{4}$ | $P(P\|P) = 0$ | $P(\langle/S\rangle\|P) = 0$ |

(a) Transition probabilities

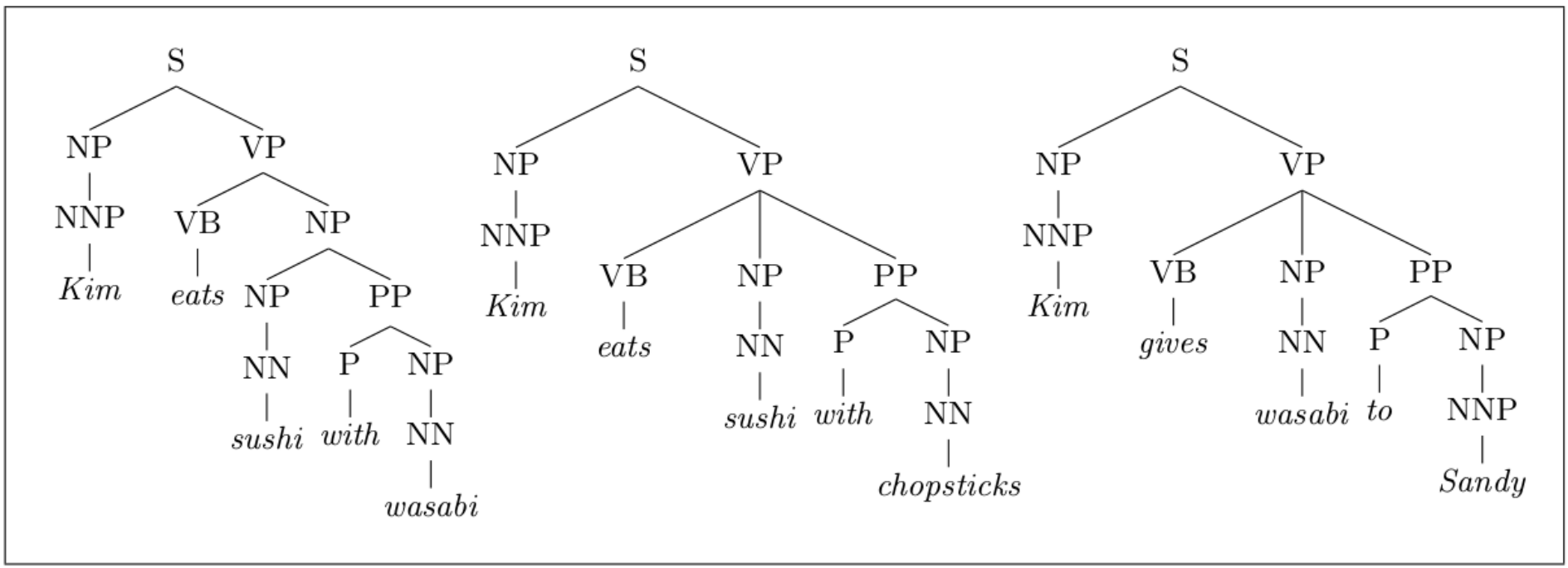
| | | | | |
|---|---|---|---|---|
| $P(\textit{flies}\|N) = \frac{1}{4}$ | $P(\textit{cats}\|N) = \frac{2}{4}$ | $P(\textit{fruit}\|N) = \frac{1}{4}$ | $P(\textit{like}\|N) = 0$ | $P(\textit{as}\|N) = 0$ |
| $P(\textit{flies}\|V) = \frac{1}{3}$ | $P(\textit{cats}\|V) = 0$ | $P(\textit{fruit}\|V) = 0$ | $P(\textit{like}\|V) = \frac{2}{3}$ | $P(\textit{as}\|V) = 0$ |
| $P(\textit{flies}\|P) = 0$ | $P(\textit{cats}\|P) = 0$ | $P(\textit{fruit}\|P) = 0$ | $P(\textit{like}\|P) = \frac{1}{4}$ | $P(\textit{as}\|P) = \frac{3}{4}$ |

(b) Emission probabilities

# (4) Hierarchical Structures

- Given the following treebank, list the rules necessary to derive the trees, their counts across all the trees and the maximum likelihood estimation of their conditional probabilities. *[7 points]*
- In a few sentences, explain the fundamental rule of chart parsing, including its purpose, the structures it operates over, and the features they have. *[9 points]*
- In a few sentences, discuss the concept of local ambiguity in parsing. What process does our generalised chart parser use for efficiently recording local ambiguities? Briefly sketch out the ideas behind the implementation of this process. *[9 points]*

# Treebank for MLE calculations



# (5) Structured Probabilistic Modelling

- What is dynamic programming and why is it used? What properties of a problem make it suitable to use a dynamic programming algorithm to solve it? *[5 points]*
- ~~Write 3–4 sentences about the differences between generative and discriminative models, including the advantages and disadvantages of each. *[4 points]*~~