IN5550: Neural Methods in Natural Language Processing

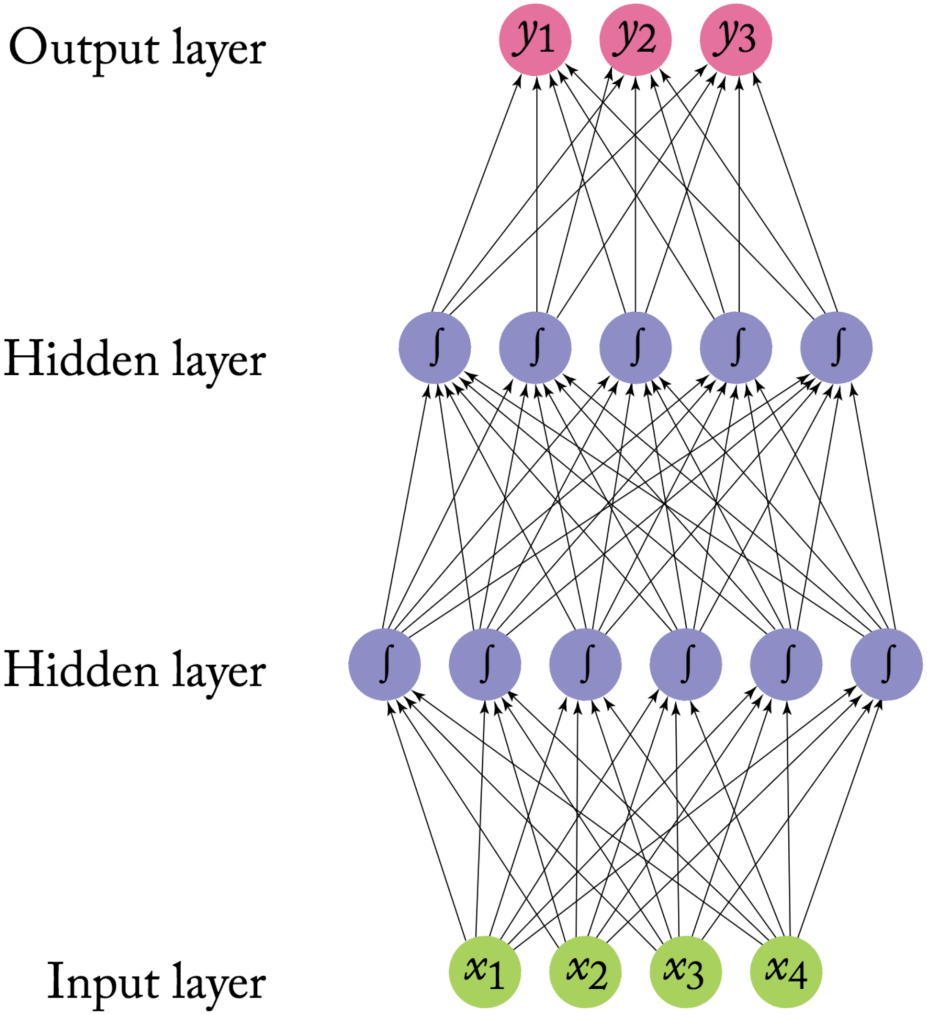
Introduction

Andrey Kutuzov, Stephan Oepen, Lilja Øvrelid,
Vinit Ravishankar, Erik Velldal, & you

University of Oslo

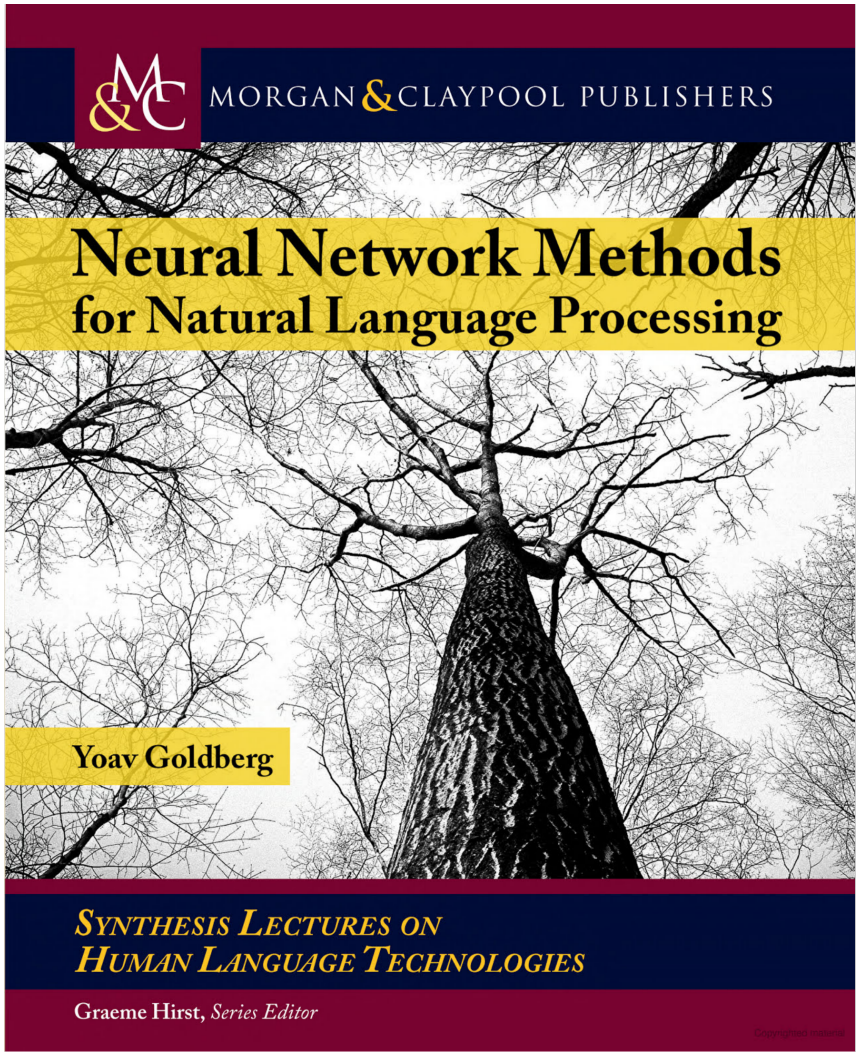
January 17, 2019

What is a neural model?



- NNs are a family of powerful machine learning models.
- Weakly based on the metaphor of a neuron.
- Learns not only to make predictions, but how to represent the data:
- non-linear transformations of the input in the 'hidden layers'.
- 'Deep Learning': NNs with several hidden layers.

Textbook



- *Neural Network Methods for Natural Language Processing* by Yoav Goldberg (Morgan & Claypool Publishers, 2017).
- Free e-version available through UiO; `http://oria.no/`
- Supplementary research papers will be added.

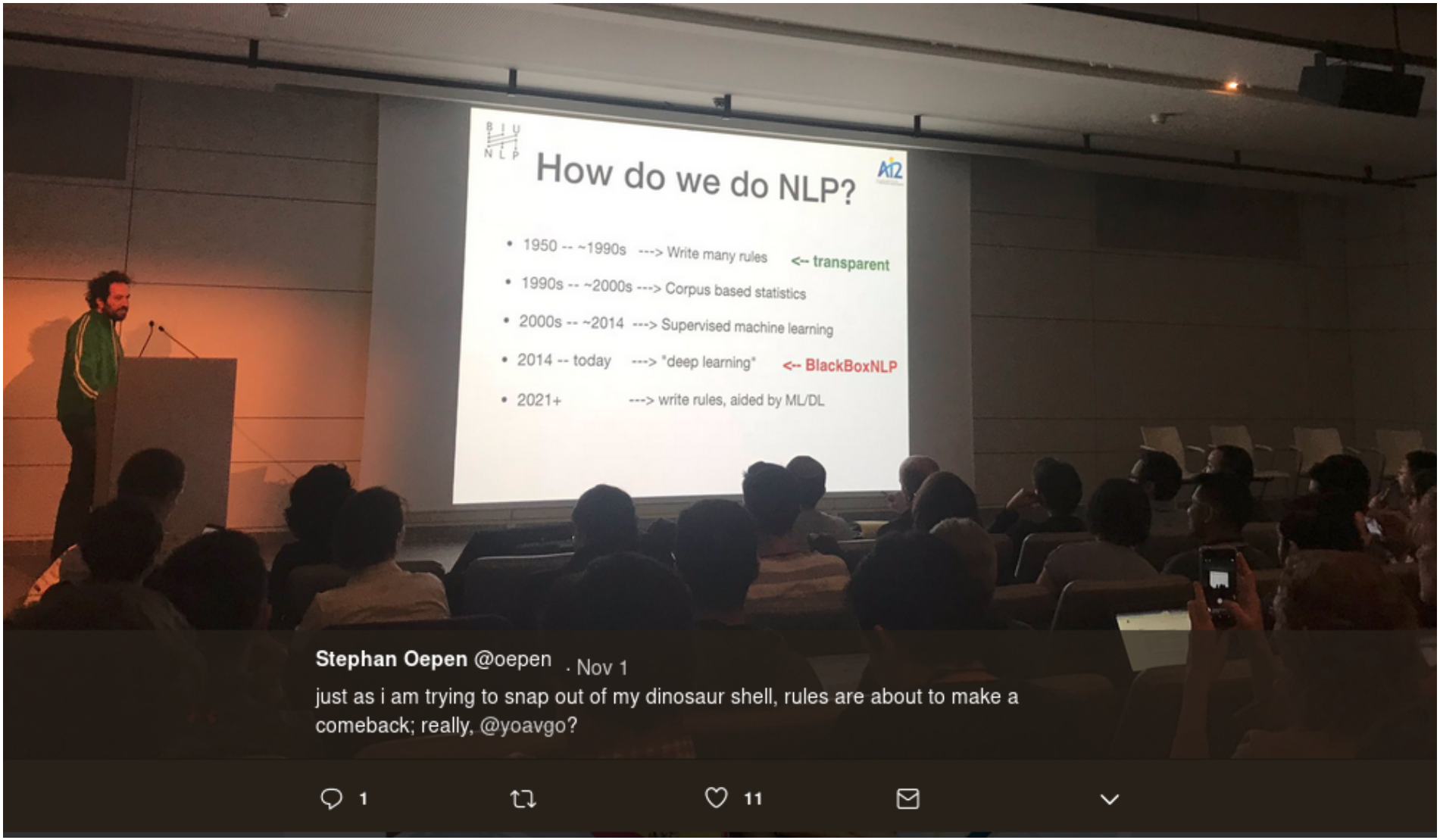
Today

- Introduction
- Why a course on neural methods?
  - Motivation
  - Historical trends
  - Success stories
  - Some contrasts between NNs and traditional ML
- Course overview
  - Lab sessions and obligatory assignments
  - Programming environment

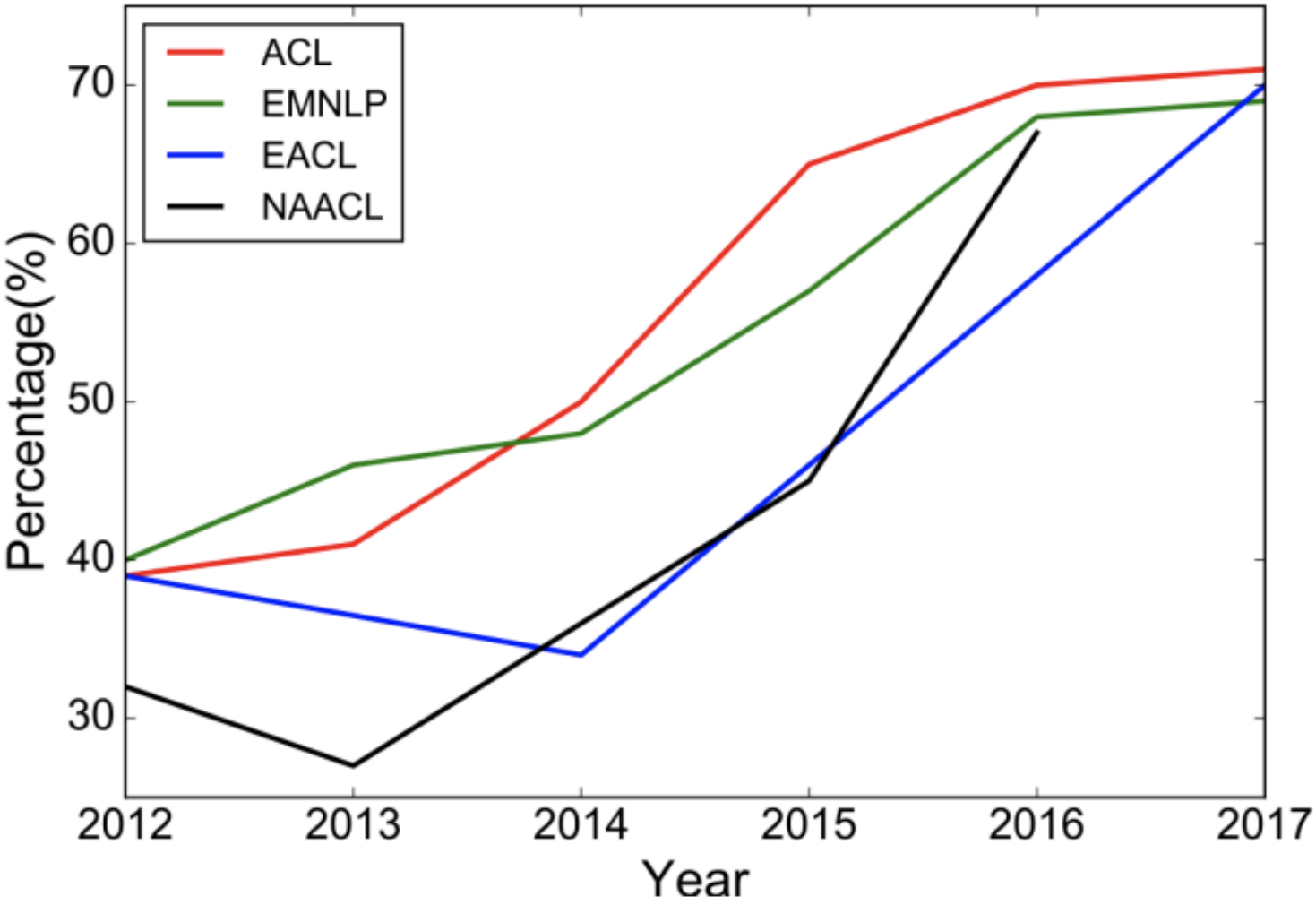
Paradigm shifts in NLP (and AI at large)

- 50s–80s: mostly rule-based (symbolic / rationalist) approaches.
- Hand-crafted formal rules and manually encoded knowledge.
- (Though some AI research on neural networks in the 40s and 50s).
- Late 80s: success with statistical ('empirical') methods in the fields of speech recognition and machine translation.
- Late 90s: NLP (and AI at large) sees a massive shift towards statistical methods and machine-learning.
- Based on automatically inferring statistical patterns from data.
- 00s: Machine-learning methods dominant.
- 2010–: neural methods increasingly replacing traditional ML.
- A revival of techniques first considered in the 40s and 50s,
- but recent developments in computational power and availability of data have given great breakthroughs in scalability and accuracy.

As seen by Yoav Goldberg



Success stories



(Young et al. (2018): *Recent Trends in Deep Learning Based Natural Language Processing*)

Success stories

- *Natural Language Processing (almost) from Scratch* by Ronan Collobert et al., 2011.
- Close to or better than SOTA for several core NLP tasks (PoS tagging, chunking, NER, and SRL).
- Pioneered much of the work on NNs for NLP.
- Cited 3903 times, as of January 2019.
- Still very influential, won test-of-time award at ICML 2018
- NNs have since been successfully applied to most NLP tasks.

Success stories

Machine translation (Google Translate)

Success stories

Machine translation (Google Translate)

- No 1:
  Kilimanjaro is a snow-covered mountain 19,710 feet high, and is said to be the highest mountain in Africa. Its western summit is called the Masai “Ngaje Ngai,” the House of God. Close to the western summit there is the dried and frozen carcass of a leopard. No one has explained what the leopard was seeking at that altitude.
- No 2:
  Kilimanjaro is a mountain of 19,710 feet covered with snow and is said to be the highest mountain in Africa. The summit of the west is called “Ngaje Ngai” in Masai, the house of God. Near the top of the west there is a dry and frozen dead body of leopard. No one has ever explained what leopard wanted at that altitude.

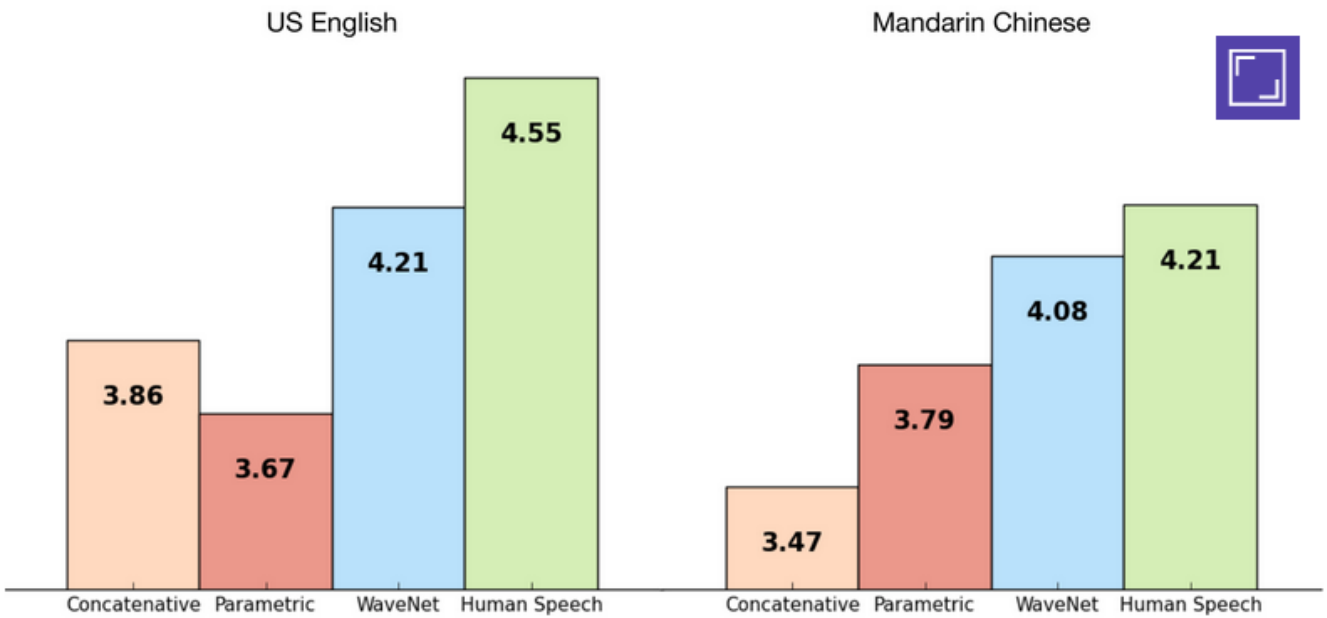
Success stories

Machine translation (Google Translate)

- No 3:
  Kilimanjaro is 19,710 feet of the mountain covered with snow, and it is said that the highest mountain in Africa. Top of the west, “Ngaje Ngai” in the Maasai language, has been referred to as the house of God. The top close to the west, there is a dry, frozen carcass of a leopard. Whether the leopard had what the demand at that altitude, there is no that nobody explained.

Success stories

Text-to-Speech (van den Oord et al., 2016):



(`https://deepmind.com/blog/wavenet-generative-model-raw-audio/`)

Success stories

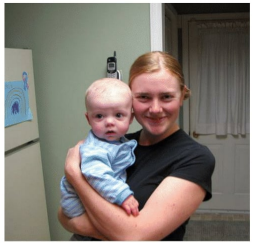
- Neural models have caused great advances in the field of image processing
- New tasks combining image and language are emerging
- Visual Question Answering:

Who is wearing glasses?

man woman

Where is the child sitting?

fridge arms



Is the umbrella upside down?

yes no

How many children are in the bed?

2 1

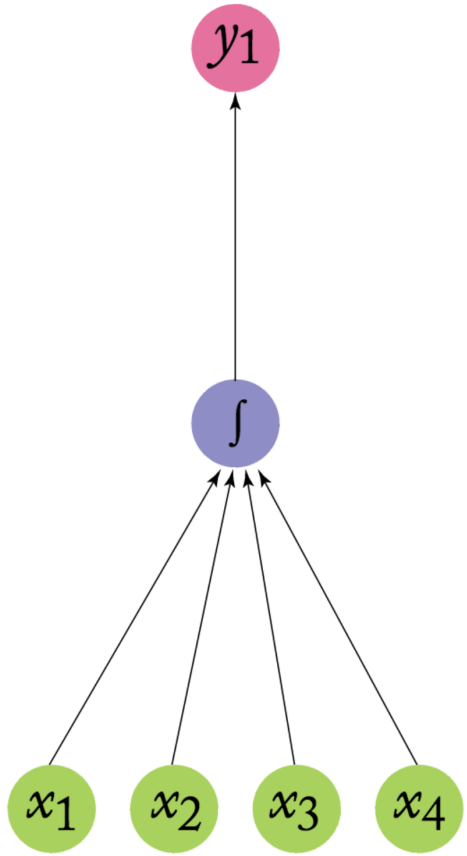
(`http://visualqa.org/`)

Contrasting NN and non-NN ML

- We will briefly review:
- Issues when working with language data and
- issues with non-neural ML,
- and how NNs can help.
- Feature engineering and model design
- The role of the designer (you).

What is a classifier?

- Very high-level:
- A learned mapping from inputs to outputs.
- Learn from labeled examples; a set of objects with correct class labels.
- 1st step in creating a classifier; defining a representation of the input!
- Typically given as a feature vector.



Feature engineering

- The art of designing features for representing objects to a classifier.
- Manually defining feature templates (for automatic feature extraction).
- Typically also involves large-scale empirical tuning to identify the best-performing configuration.
- Although there is much overlap in the types of features used across tasks, performance is highly dependent on the specific task and dataset.
- We will review some of the most standard feature types...

‘Atomic’ features

- The word forms occurring in the target context (e.g. document, sentence, or window).
- E.g., *Bag-of-words* (BoW): All words within the context, *unordered*.

‘The sandwiches were hardly fresh and the service not impressive.’

$$\Rightarrow$$

{service, fresh, sandwiches, impressive, not, hardly, . . . }

‘Atomic’ features

- The word forms occurring in the target context (e.g. document, sentence, or window).
- E.g., *Bag-of-words* (BoW): All words within the context, *unordered*.

‘The sandwiches were hardly fresh and the service not impressive.’

$$\Rightarrow$$

{service, fresh, sandwiches, impressive, not, hardly, . . . }

- Can also be *ordered* (positional).
- Feature vectors typically record (some function of) frequency counts.
- Each dimension encodes one feature (e.g., co-occurrence with ‘fresh’).

A bit more linguistically informed

- Various levels of pre-processing often performed to infer more linguistically informed features, e.g.:
- Lemmatization
- Part-of-speech (PoS) tagging
- 'Chunking' (phrase-level / shallow parsing)
- Often need to define combined features to capture relevant information.
- E.g: BoW of lemmas + PoS (*sandwich_NOUN*)

Dealing with compositionality

man bites dog $\neq$ *dog bites man*

- Some complex feature combinations attempt to take account of the fact that language is compositional.
- E.g. by applying parsing to infer information about syntactic and semantic relations between the words.

Dealing with compositionality

man bites dog ≠ *dog bites man*

- Some complex feature combinations attempt to take account of the fact that language is compositional.
- E.g. by applying parsing to infer information about syntactic and semantic relations between the words.
- A more simplistic approximation that is often used in practice:
- n-grams (typically bigrams and trigrams).

{service, fresh, sandwiches, impressive, not, hardly, . . . }

vs.

{'hardly fresh', 'not impressive', 'service not', . . . }

- (The need for combined features can also be related to the linearity of a model; we return to this later in the course.)

Discreteness and data sparseness

- The resulting feature vectors are very high-dimensional; typically in the order of thousands or even millions (!) of dimensions.
- Very sparse; only a very small ratio of non-zero features.

Discreteness and data sparseness

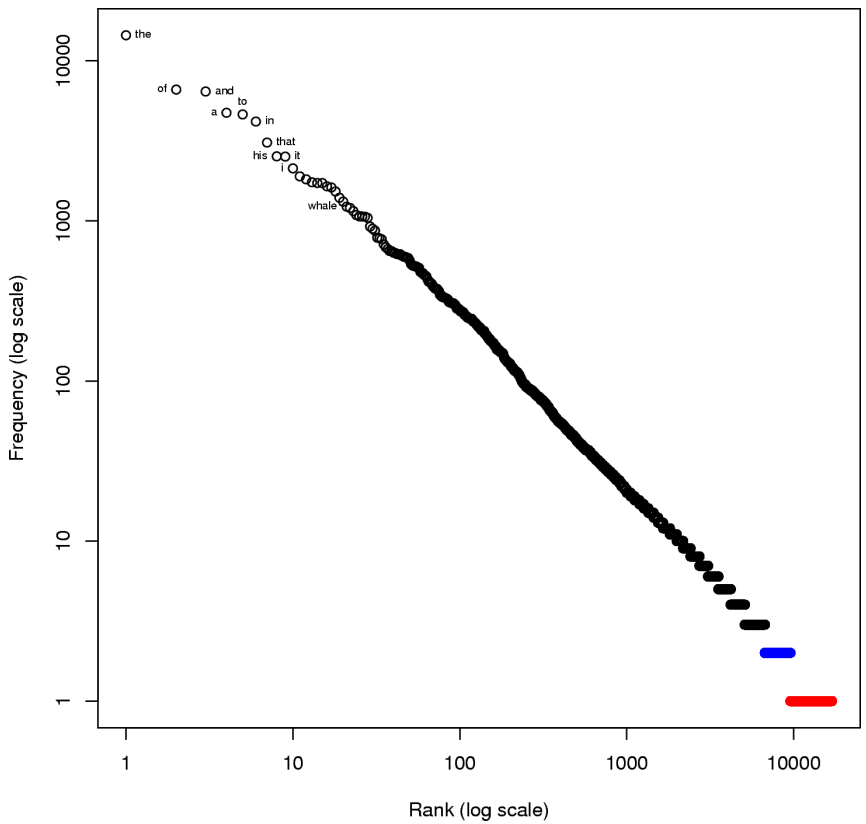
- The resulting feature vectors are very high-dimensional; typically in the order of thousands or even millions (!) of dimensions.
- Very sparse; only a very small ratio of non-zero features.
- The features we have considered are discrete and categorical.
- Categorical features that are all equally distinct.
- No sharing of information;
- In our representation, a feature recording the presence of 'impressive' is completely unrelated to 'awesome', 'admirable', etc.

Discreteness and data sparseness

- The resulting feature vectors are very high-dimensional; typically in the order of thousands or even millions (!) of dimensions.
- Very sparse; only a very small ratio of non-zero features.
- The features we have considered are discrete and categorical.
- Categorical features that are all equally distinct.
- No sharing of information;
- In our representation, a feature recording the presence of ‘impressive’ is completely unrelated to ‘awesome’, ‘admirable’, etc.
- Made worse by the ubiquitous problem of data sparseness.

Data sparseness

- Language use is creative and productive:
- No corpus can be large enough to provide full coverage.
- Zipf's law and the long tail.



- Word types in *Moby Dick*
- 44% occur only once (red)
- 17% occur twice (blue)
- *the* = 7% of the tokens

- On top of this, the size of our data is often limited by our need for labeled training data.

Alleviating the problems of discreteness and sparseness

- Linguistic pre-processing can alleviate the problems to *some* degree.
- E.g. fewer unique lemmas than full-forms; PoS provides some generalization; etc.
- Other class-based features sometimes available:
- E.g. dictionary resources like Gazetteers (for NER) or WordNet.
- Add features corresponding to the ID of WordNet synsets or hypernyms.
- Provides some degree of generalization.

Alleviating the problems of discreteness and sparseness

- Linguistic pre-processing can alleviate the problems to *some* degree.
- E.g. fewer unique lemmas than full-forms; PoS provides some generalization; etc.
- Other class-based features sometimes available:
- E.g. dictionary resources like Gazetteers (for NER) or WordNet.
- Add features corresponding to the ID of WordNet synsets or hypernyms.
- Provides some degree of generalization.

Another angle

- We have lots of text but typically very little labeled data. . .
- How can be make better use of unlabeled data.
- Include distributional information.

Distributional information

- We can incorporate information about the similarities between our discrete features by considering distributional information.
- The distributional hypothesis: words that occur in similar contexts are similar (syntactically or semantically).
- How can we record and represent the contextual distribution of words?

Distributional information

- We can incorporate information about the similarities between our discrete features by considering distributional information.
- The distributional hypothesis: words that occur in similar contexts are similar (syntactically or semantically).
- How can we record and represent the contextual distribution of words?
- Summing feature vectors (like the ones we've discussed today) for all occurrences of a given word gives us a distributional word vector!
- Vector distance indicate word similarity.
- Completely unsupervised; can be generated from unlabeled data.
- Great, but how can we incorporate this into our model?

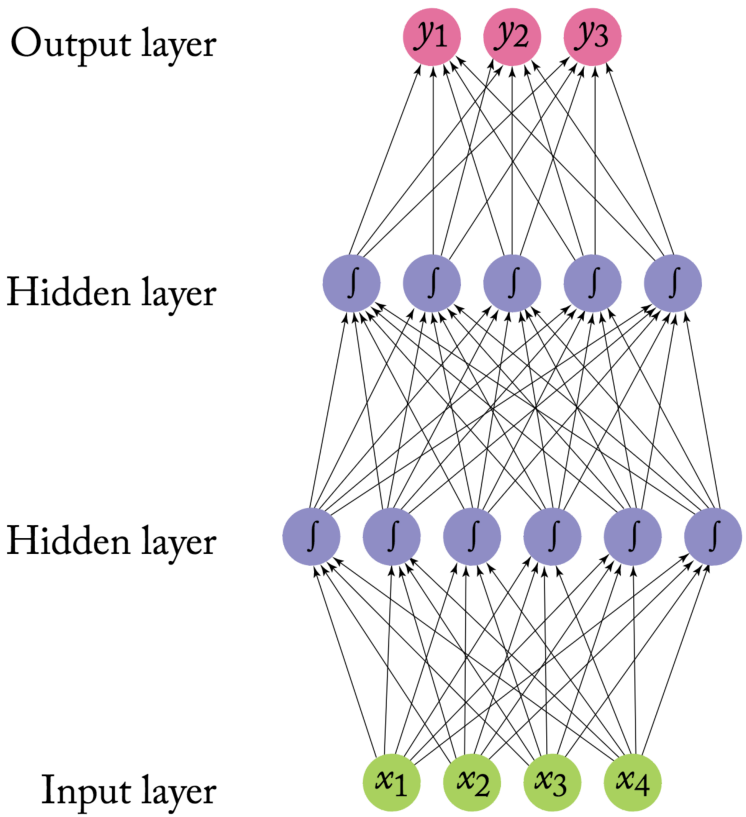
Distributional word vectors

- Can be incorporated in a model either by clustering the vectors (also unsupervised) and using cluster ids as class-based features,
- or by using the distributional vectors to replace the original discrete features (e.g. by concatenation or summing).
- In both cases it means that even for words not seen during training the model can have learned useful information about it!

Word embeddings

- A particular type of distributional word vector:
- Mapped onto a dense and low-dimensional space (typically 50–300 d.)
- Makes them well-suited for replacing discrete features.
- Not just distributional, but distributed.
- Popular toolkits: word2vec, fastText, GloVe, . . .
- Andrey will be covering word embeddings in lectures 4–6.
- The most common input representation to NNs for NLP tasks.
- Can also be learned from scratch by the NN itself.
- More abstract feature representations are then learned automatically by the network (in the form of hidden layers).
- Representation learning

Some pros and cons of NNs



NNs hold promise to...

- reduce manual feature engineering,
- make better use of unlabeled data,
- through the use of distributional continuous input representations,
- and further learn more task-adapted internal representations of the data,
- giving less language-specific models.
- Tends to scale better with more data.
- Though at the cost of being less interpretable and more complex with more parameters.

Focus of manual engineering

Traditional ML

- The model architecture is given.
- Focus on designing and tuning the features.
- Input: high-dimensional and sparse; manually defined features.

Neural methods

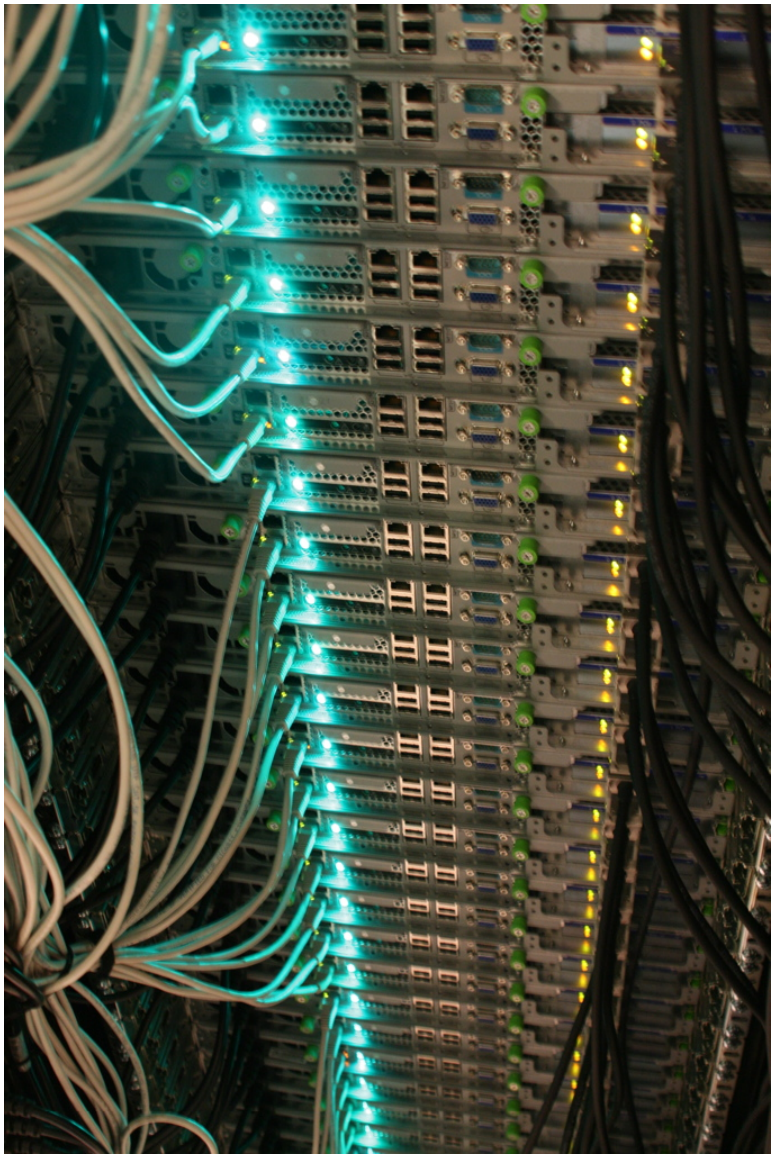
- Uniform input representation: word embeddings.
- Focus on designing and tuning the model architecture.
- Input: low-dimensional and dense; learned unsupervised. The network learns additional internal representations.

- No free lunch...

Getting to Know Each Others

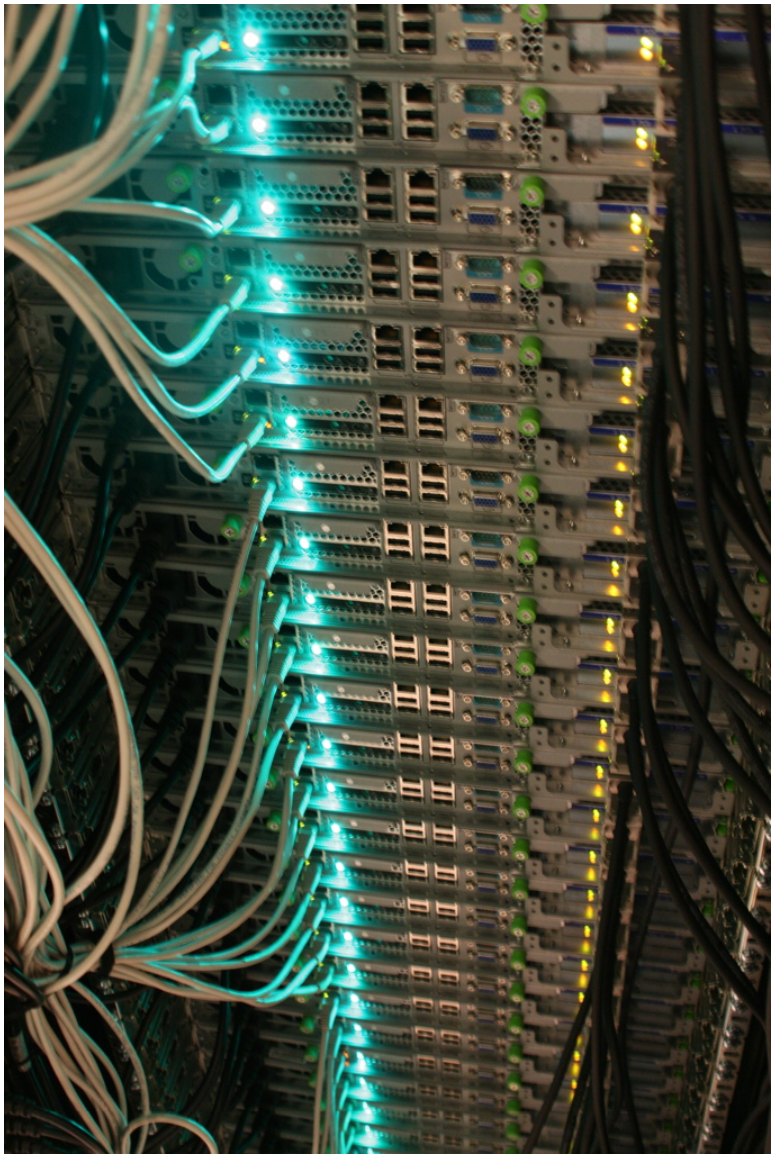
https://skjema.uio.no/110217

Scientific Programming in IN5550



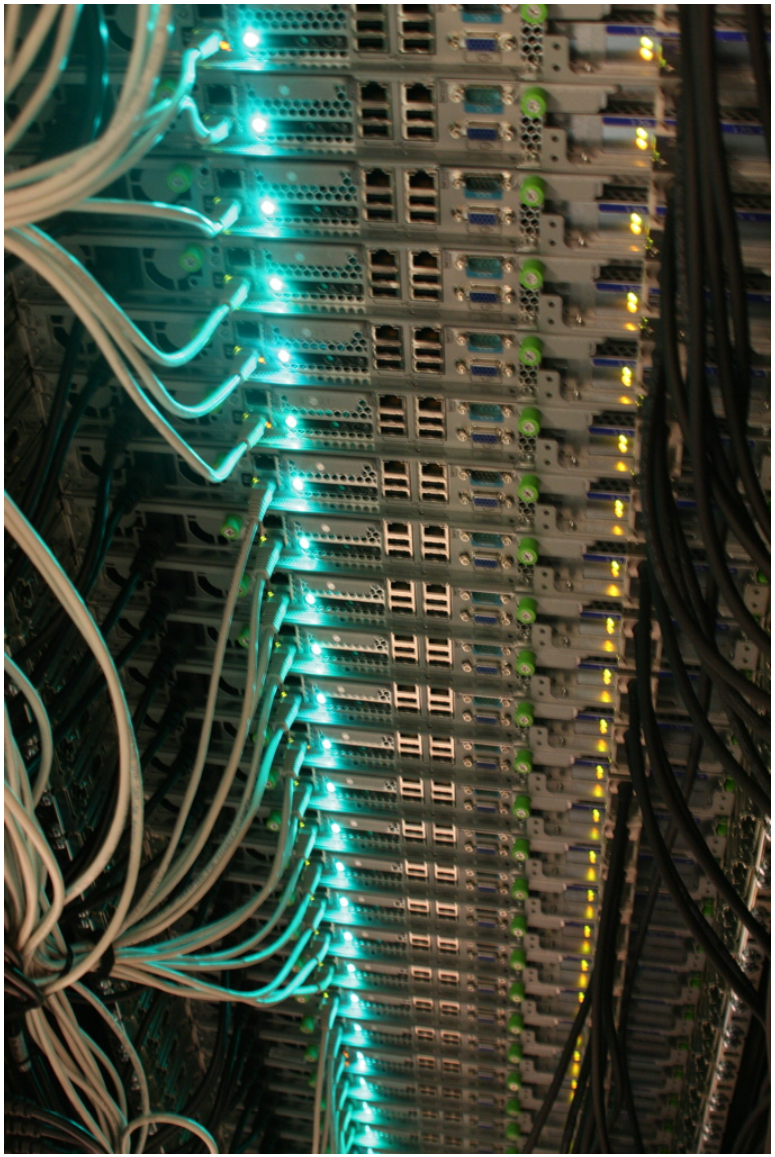
- Strong practical and programming elements in this course;

Scientific Programming in IN5550



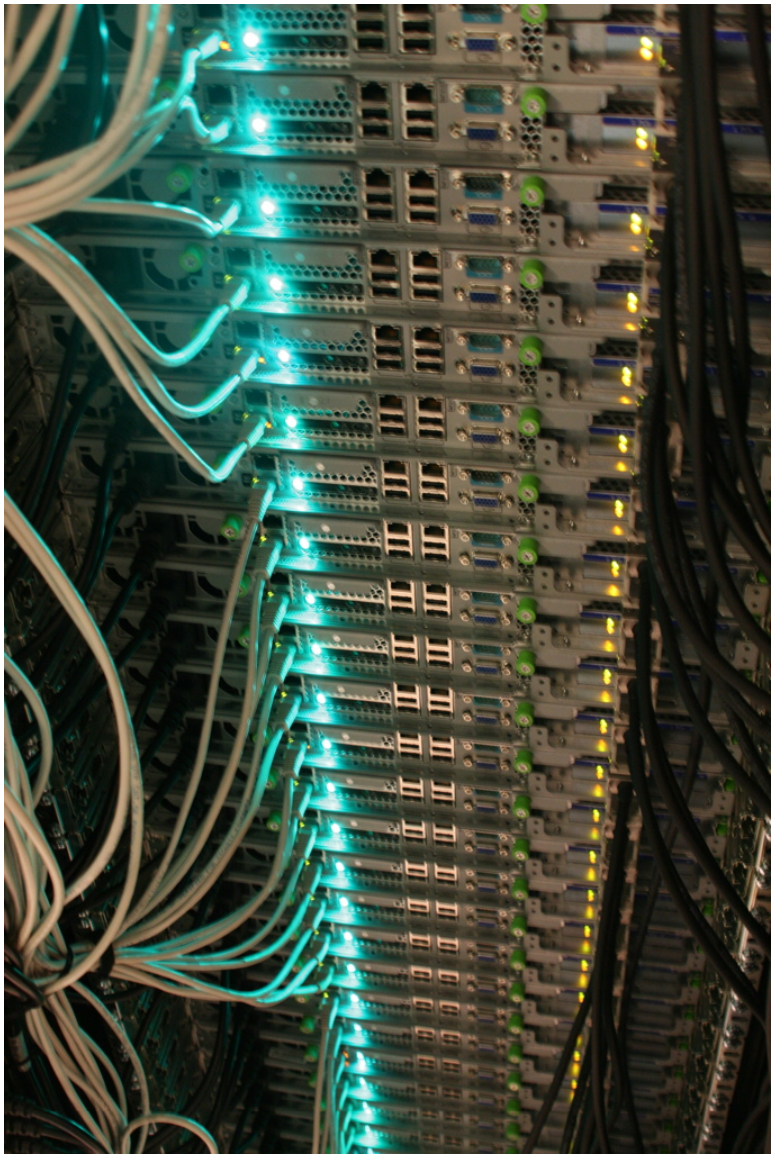
- Strong practical and programming elements in this course;
- three obligatory assignments; all predominantly ‘hands-on’;

Scientific Programming in IN5550



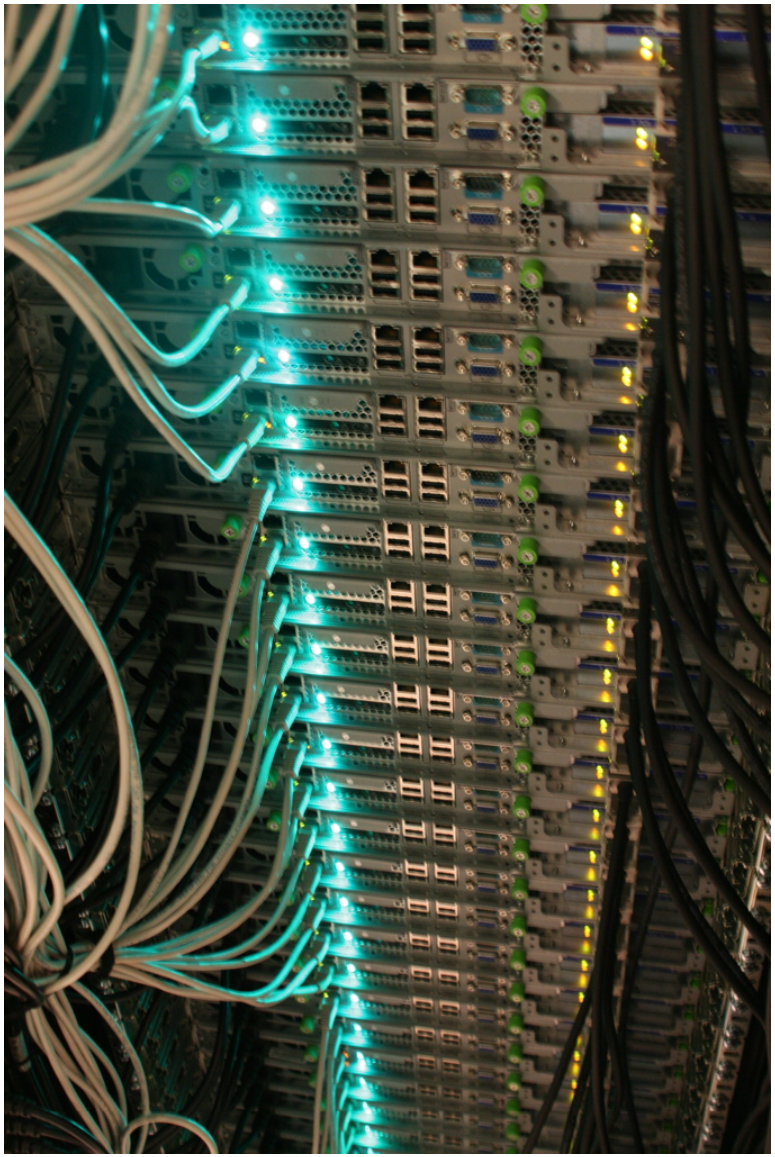
- Strong practical and programming elements in this course;
- three obligatory assignments; all predominantly ‘hands-on’;
- representative of sub-problems in typical MSc theses at LTG;

Scientific Programming in IN5550



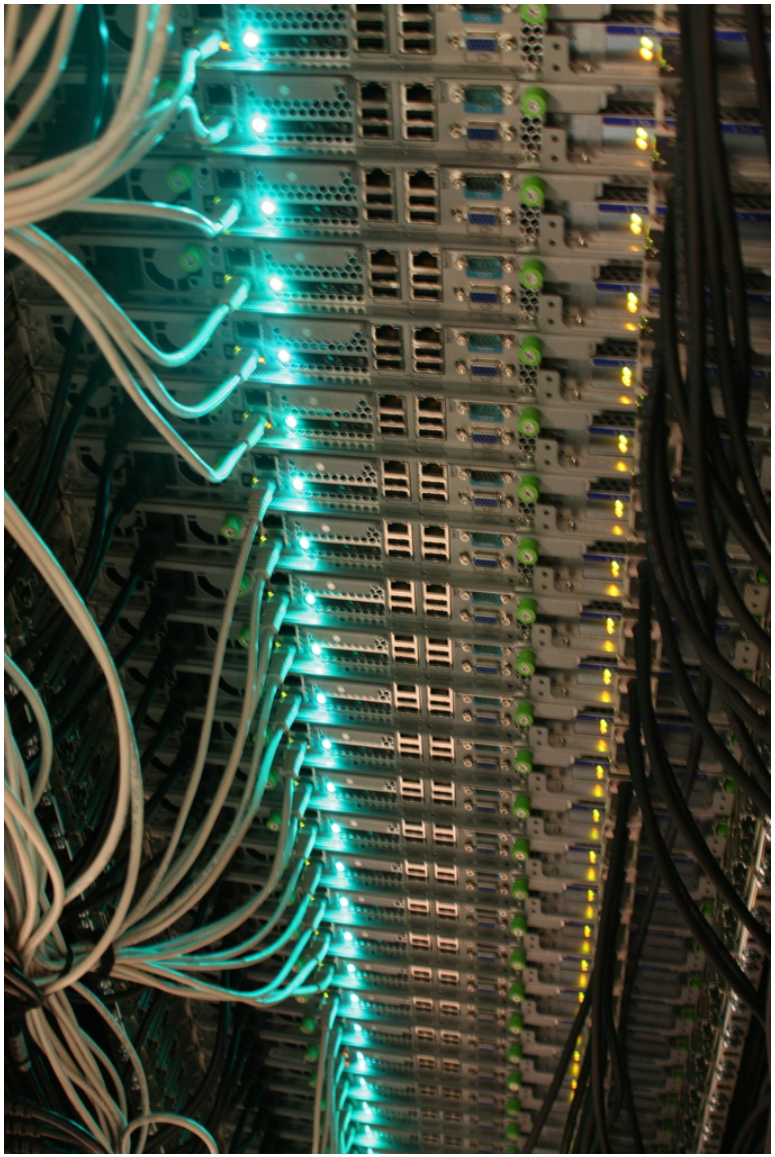
- Strong practical and programming elements in this course;
- three obligatory assignments; all predominantly ‘hands-on’;
- representative of sub-problems in typical MSc theses at LTG;
- fairly data- and compute-intensive throughout the semester;

Scientific Programming in IN5550



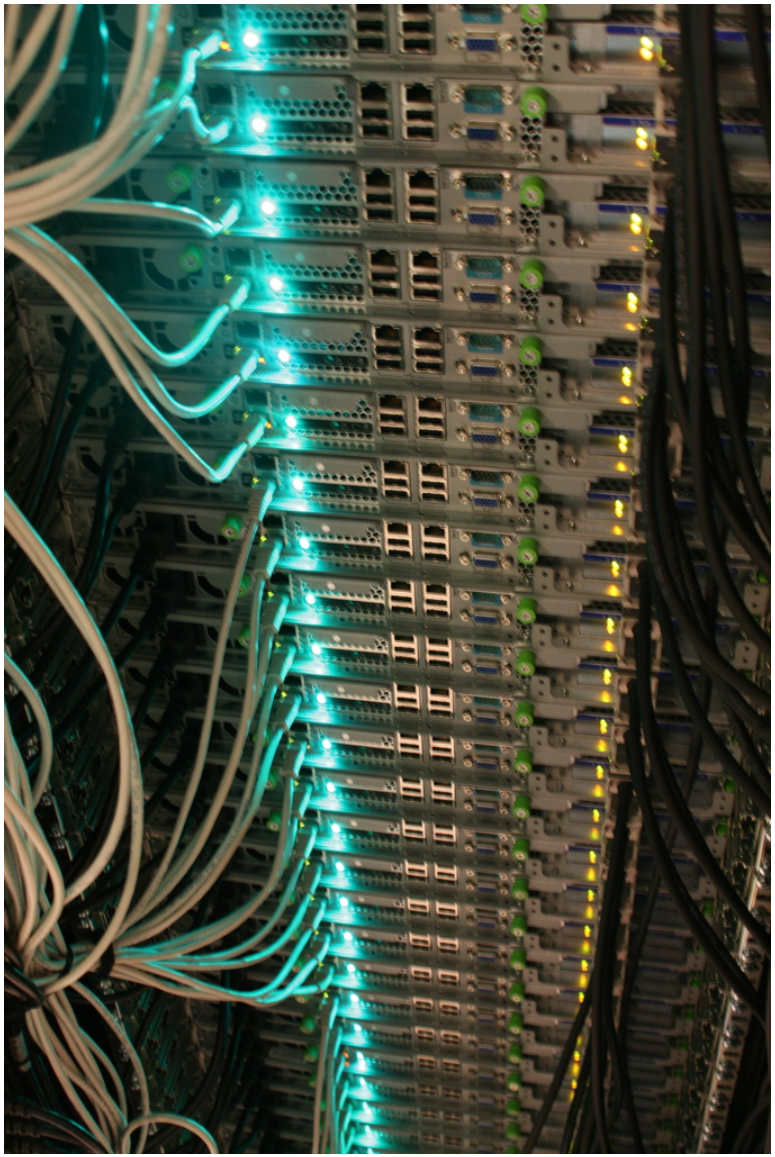
- Strong practical and programming elements in this course;
- three obligatory assignments; all predominantly ‘hands-on’;
- representative of sub-problems in typical MSc theses at LTG;
- fairly data- and compute-intensive throughout the semester;
- learn how to use the national Abel supercluster

Scientific Programming in IN5550



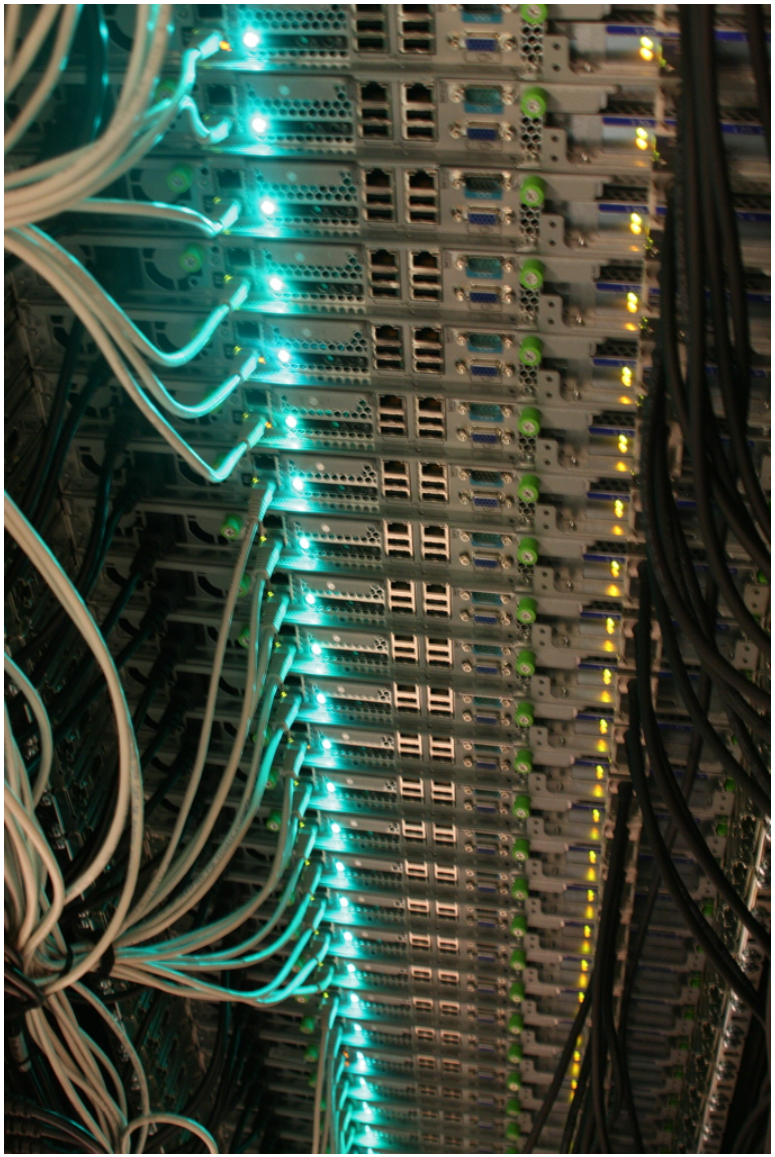
- Strong practical and programming elements in this course;
- three obligatory assignments; all predominantly 'hands-on';
- representative of sub-problems in typical MSc theses at LTG;
- fairly data- and compute-intensive throughout the semester;
- learn how to use the national Abel supercluster (10,000$^+$ cpus);

Scientific Programming in IN5550



- Strong practical and programming elements in this course;
- three obligatory assignments; all predominantly 'hands-on';
- representative of sub-problems in typical MSc theses at LTG;
- fairly data- and compute-intensive throughout the semester;
- learn how to use the national Abel supercluster (10,000$^+$ cpus);
- ranked 96 among the world's fastest supercomputers

Scientific Programming in IN5550



- Strong practical and programming elements in this course;
- three obligatory assignments; all predominantly 'hands-on';
- representative of sub-problems in typical MSc theses at LTG;
- fairly data- and compute-intensive throughout the semester;
- learn how to use the national Abel supercluster (10,000$^+$ cpus);
- ranked 96 among the world's fastest supercomputers (in 2012).

Deep Learning through Python

- Python is a simple Lisp dialect (with an idiosyncratic syntax) with great popularity for neural network–based machine learning;

Deep Learning through Python

- Python is a simple Lisp dialect (with an idiosyncratic syntax) with great popularity for neural network–based machine learning;
- it provides a very convenient, high-level scripting language with a gentle learning curve; works easily across different platforms;

Deep Learning through Python

- Python is a simple Lisp dialect (with an idiosyncratic syntax) with great popularity for neural network–based machine learning;
- it provides a very convenient, high-level scripting language with a gentle learning curve; works easily across different platforms;
- comprehensive standard library; ecosystem of community-maintained add-on modules with specialized (and optimized) functionality;

Deep Learning through Python

- Python is a simple Lisp dialect (with an idiosyncratic syntax) with great popularity for neural network–based machine learning;
- it provides a very convenient, high-level scripting language with a gentle learning curve; works easily across different platforms;
- comprehensive standard library; ecosystem of community-maintained add-on modules with specialized (and optimized) functionality;
- pretty much everything open-source; we provide reference environment on Abel; in principle possible to install 'at home' (for development).

A Menagerie of Interoperable Modules

- NumPy for efficient multi-dimensional arrays (aka ‘tensors’) and general linear algebra;

A Menagerie of Interoperable Modules

- NumPy for efficient multi-dimensional arrays (aka ‘tensors’) and general linear algebra;
- Any self-respecting technology giant today develops their own DL framework;
- plus a few from university environments;
- open source $\rightarrow$ community involvement;

A Menagerie of Interoperable Modules

- NumPy for efficient multi-dimensional arrays (aka ‘tensors’) and general linear algebra;
- Any self-respecting technology giant today develops their own DL framework;
- plus a few from university environments;
- open source $\rightarrow$ community involvement;
- PyTorch (Facebook) is a mature software infrastructure (built natively in C^{++});

A Menagerie of Interoperable Modules

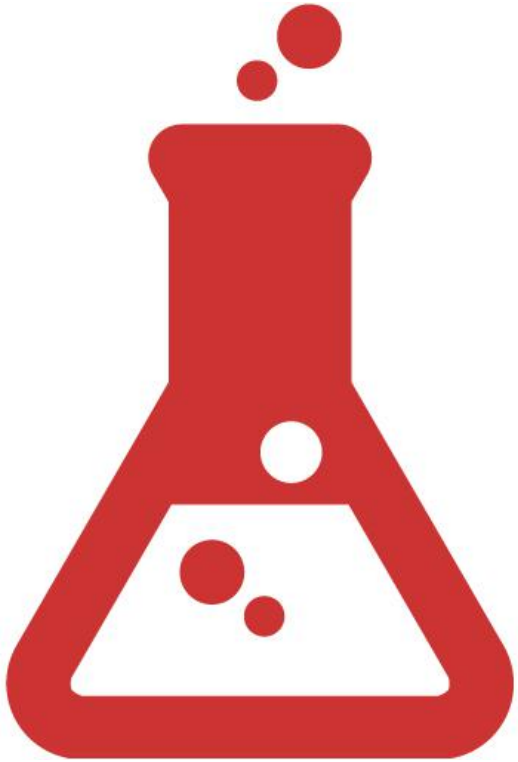
- NumPy for efficient multi-dimensional arrays (aka ‘tensors’) and general linear algebra;
- Any self-respecting technology giant today develops their own DL framework;
- plus a few from university environments;
- open source $\rightarrow$ community involvement;
- PyTorch (Facebook) is a mature software infrastructure (built natively in C^{++});
- Gensim for (large-scale) distributional analysis $\rightarrow$ (word) ‘embeddings’;

A Menagerie of Interoperable Modules

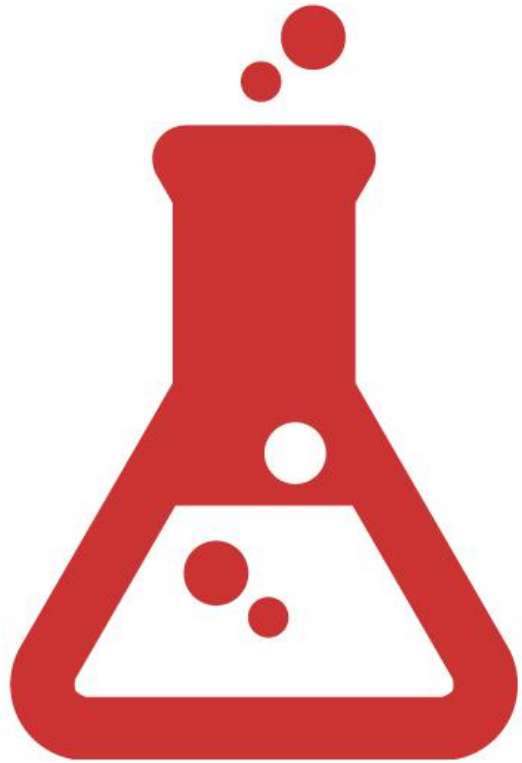
- NumPy for efficient multi-dimensional arrays (aka ‘tensors’) and general linear algebra;
- Any self-respecting technology giant today develops their own DL framework;
- plus a few from university environments;
- open source $\rightarrow$ community involvement;
- PyTorch (Facebook) is a mature software infrastructure (built natively in C^{++});
- Gensim for (large-scale) distributional analysis $\rightarrow$ (word) ‘embeddings’;
- all integrated in a course-specific Python 3 installation on Abel (see the course page).

Labs and Obligatory Assignments (*grupper & obliger*)

- Lab: Tuesdays, 14:15–16:00;
- three obligatory assignments;
- rigid schedule; all assignments must be submitted;
- minimum 60% of points across all three required to qualify for exam;
- no re-submissions.



Labs and Obligatory Assignments (*grupper & obliger*)



- Lab: Tuesdays, 14:15–16:00;
- three obligatory assignments;
- rigid schedule; all assignments must be submitted;
- minimum 60% of points across all three required to qualify for exam;
- no re-submissions.

Mechanics

- Schedule: `https://www.uio.no/studier/emner/matnat/ifi/IN5550/v19/assignments.html`
- Group work encouraged (1–3 students); please give it a shot!
- Final exam as 'baby' MSc project in April–May; team work possible.

Course Communication

- Questions?

Course Communication

- Questions?
  - Piazza: on-line discussion board linked from course page.
  - `in5550-help@ifi.uio.no` reaches all course staff.
  - Microsoft GitHub: `https://github.uio.no/in5550/2019`.

Course Communication

- Questions?
  - Piazza: on-line discussion board linked from course page.
  - `in5550-help@ifi.uio.no` reaches all course staff.
  - Microsoft GitHub: `https://github.uio.no/in5550/2019`.
- Messages:
  - Read (or forward) your UiO email regularly;
  - Check the course page and schedule regularly;
  - Participate in the on-line discussion board.

Next week

- Introducing mathematical notation for describing classifiers.
- From linear regression to feed-forward networks and multi-layer perceptrons.
- Linear vs non-linear decision boundaries