Natural Language Processing: Part II Overview of Natural Language Processing (L90): ACS Lecture 9

Ann Copestake

Computer Laboratory
University of Cambridge

October 2018

Distributional semantics and deep learning: outline

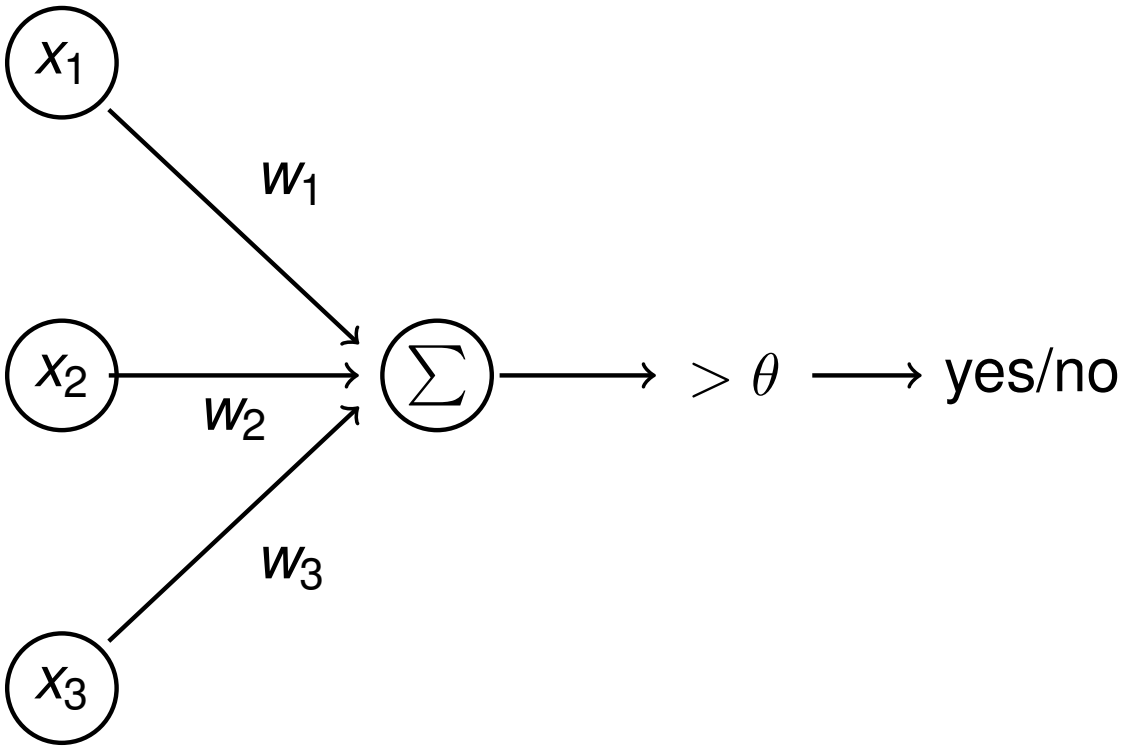
- Neural networks in pictures
- word2vec
- Visualization of NNs
- Visual question answering
- Perspective

Some slides borrowed from Aurelie Herbelot

Outline.

Perceptron

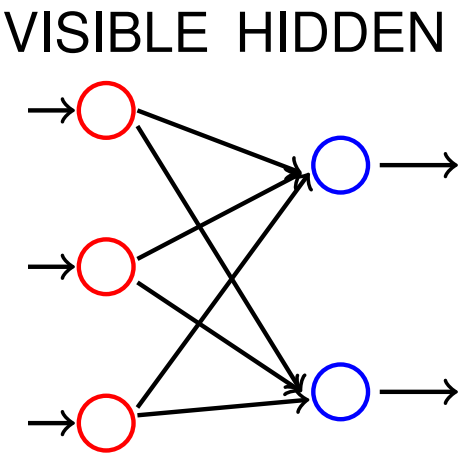
- Early model (1962): no hidden layers, just a linear classifier, summation output.



Dot product of an input vector $\vec{x}$ and a weight vector $\vec{w}$, compared to a threshold θ

Restricted Boltzmann Machines

- Boltzmann machine: hidden layer, arbitrary interconnections between units. Not effectively trainable.
- Restricted Boltzmann Machine (RBM): one input and one hidden layer, no intra-layer links.

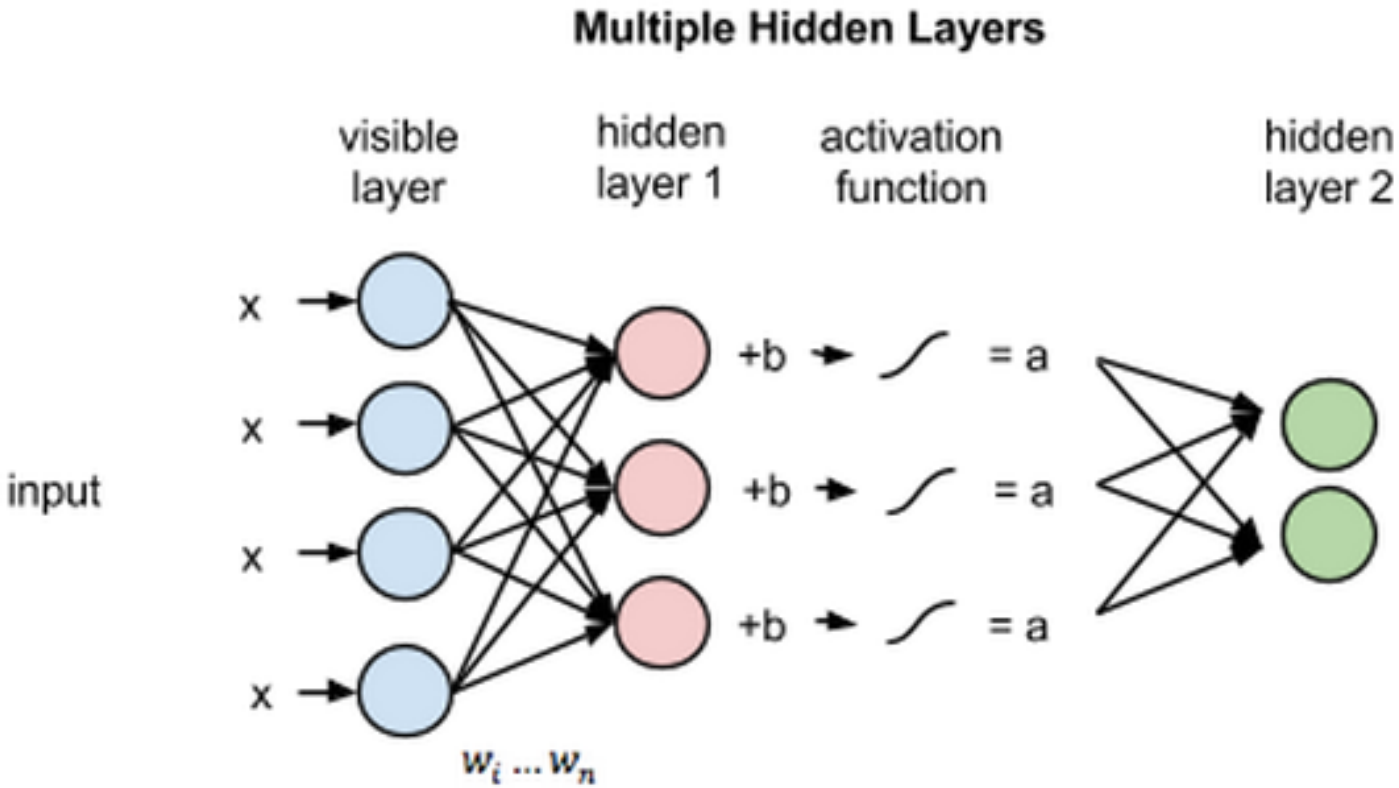


$w_1, \ldots w_6$ b (bias)

Restricted Boltzmann Machines

- Hidden layer (note one hidden layer can model arbitrary function, but not necessarily trainable).
- RBM layers allow for efficient implementation: weights can be described by a matrix, fast computation.
- One popular deep learning architecture is a combination of RBMs, so the output from one RBM is the input to the next.
- RBMs can be trained separately and then fine-tuned in combination.
- The layers allow for efficient implementations and successive approximations to concepts.

Combining RBMs: deep learning



https://deeplearning4j.org/restrictedboltzmannmachine

Sequences

- Combined RBMs etc, cannot handle sequence information well (can pass them sequences encoded as vectors, but input vectors are fixed length).
- So different architecture needed for sequences and most language and speech problems.
- RNN: Recurrent neural network.
- Long short term memory (LSTM): development of RNN, more effective for (some?) language applications.

Multimodal architectures

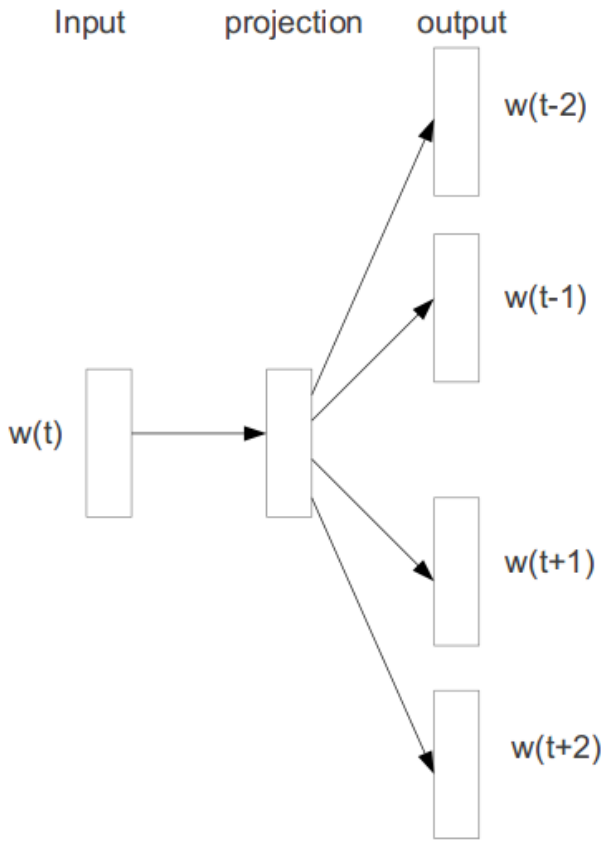
- Input to a NN is just a vector: we can combine vectors from different sources.
- e.g., features from a CNN for visual recognition concatenated with word embeddings.
- multimodal systems: captioning, visual question answering (VQA).

Outline.

Embeddings

- embeddings: distributional models with dimensionality reduction, based on prediction
- word2vec: as originally described (Mikolov et al 2013), a NN model using a two-layer network (i.e., not deep!) to perform dimensionality reduction.
- two possible architectures:
  - given some context words, predict the target (CBOW)
  - given a target word, predict the contexts (Skip-gram)
- Very computationally efficient, good all-round model (good hyperparameters already selected).

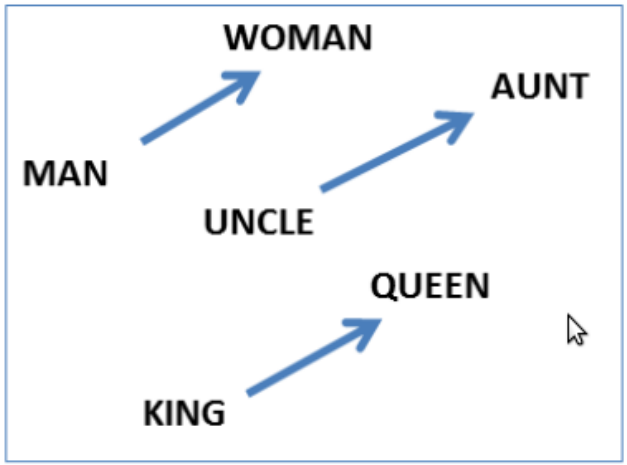
The Skip-gram model

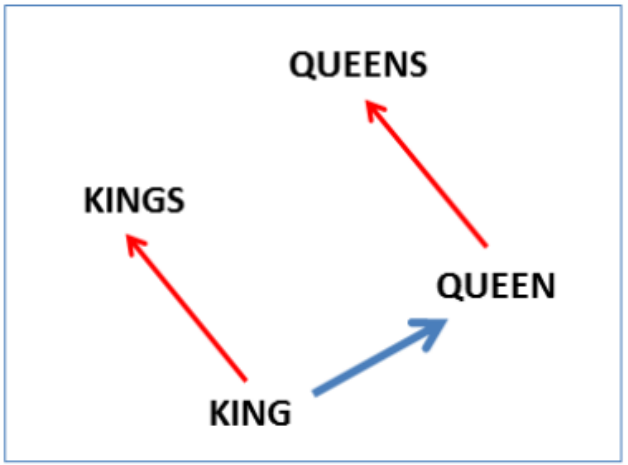


Features of word2vec representations

- A representation is learnt at the reduced dimensionality straightaway: we are outputting vectors of a chosen dimensionality (parameter of the system).
- Usually, a few hundred dimensions: dense vectors.
- The dimensions are not interpretable: it is impossible to look into ‘characteristic contexts’.
- For many tasks, word2vec (skip-gram) outperforms standard count-based vectors.
- But mainly due to the hyperparameters and these can be emulated in standard count models (see Levy et al).

What Word2Vec is famous for





BUT . . . see Levy et al and Levy and Goldberg for discussion

The actual components of word2vec

- A vocabulary. (Which words do I have in my corpus?)
- A table of word probabilities.
- Negative sampling: tell the network what *not* to predict.
- Subsampling: don't look at all words and all contexts.

Negative sampling

Instead of doing full softmax (final stage in a NN model to get probabilities, very expensive), word2vec is trained using logistic regression to discriminate between real and fake words:

- Whenever considering a word-context pair, also give the network some contexts which are not the actual observed word.
- Sample from the vocabulary. The probability to sample something more frequent in the corpus is higher.
- The number of negative samples will affect results.

Subsampling

- Instead of considering all words in the sentence, transform it by randomly removing words from it:
  considering all sentence transform randomly words
- The subsampling function makes it more likely to remove a frequent word.
- Note that word2vec does not use a stop list.
- Note that subsampling affects the window size around the target (i.e., means word2vec window size is not fixed).
- Also: weights of elements in context window vary.

Using word2vec

- predefined vectors or create your own
- can be used as input to NN model
- many researchers use the gensim Python library
  `https://radimrehurek.com/gensim/`
- Emerson and Copestake (2016) find significantly better performance on some tests using parsed data
- Levy et al's papers are very helpful in clarifying word2vec behaviour
- Bayesian version: Barkan (2016)

  `https://arxiv.org/ftp/arxiv/papers/1603/1603.06571.pdf`

doc2vec: Le and Mikolov (2014)

- Learn a vector to represent a 'document': sentence, paragraph, short document.
- skip-gram trained by predicting context word vectors given an input word, distributed bag of words (dbow) trained by predicting context words given a document vector.
- order of document words ignored, but also dmpv, analogous to cbow: sensitive to document word order
- Options:
  1. start with random word vector initialization
  2. run skip-gram first
  3. use pretrained embeddings (Lau and Baldwin, 2016)

doc2vec: Le and Mikolov (2014)

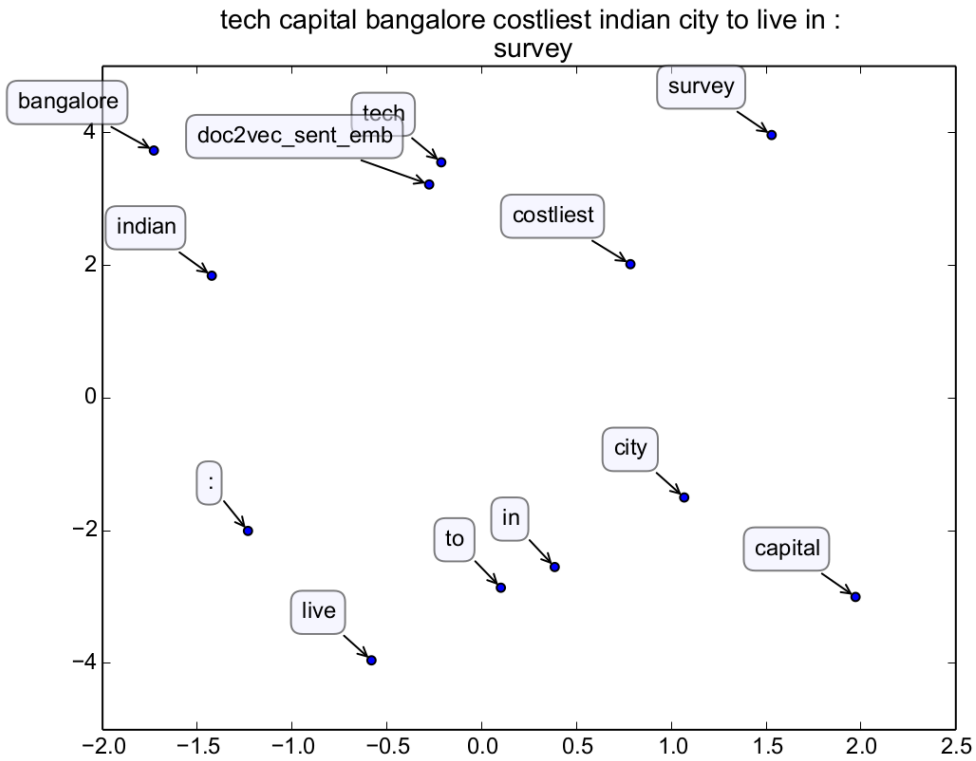
- Learned document vector effective for various tasks, including sentiment analysis.
- Lots and lots of possible parameters.
- Some initial difficulty in reproducing results, but Lau and Baldwin (2016) have a careful investigation of doc2vec, demonstrating its effectiveness.

Outline.

Finding out what NNs are really doing

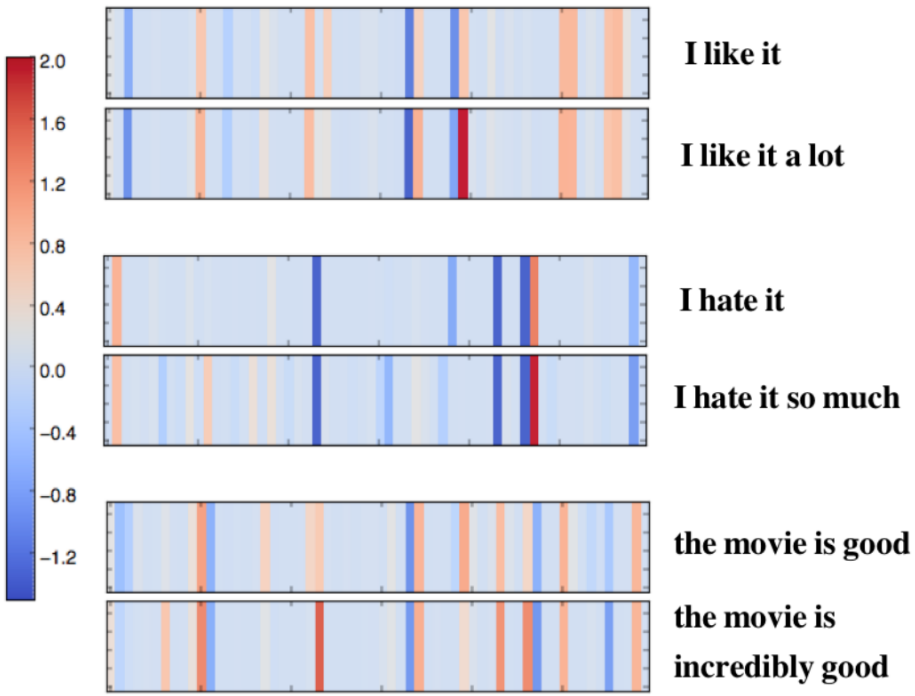
- Careful investigation of models (sometimes including going through code), describing as non-neural models (Omer Levy, word2vec).
- Building proper baselines (e.g., Zhou et al, 2015 for VQA).
- Selected and targeted experimentation.
- Visualization.

t-SNE example: Lau and Baldwin (2016



arxiv.org/abs/1607.05368

Heatmap example: Li et al (2015)



Embeddings for sentences obtained by composing embeddings for words. Heatmap shows values for dimensions.

`arxiv.org/abs/1506.01066`

Outline.

Multimodal architectures

- Input to a NN is just a vector: we can combine vectors from different sources.
- e.g., features from a CNN for visual recognition concatenated with word embeddings.
- multimodal systems: captioning, visual question answering (VQA).

Visual Question Answering

- System is given a picture and a question about the picture which it has to answer.
- Best known dataset: COCO VQA (Agrawal et al, 2016).
- Questions and answers for images from Amazon Mechanical Turk.
- Task: provide questions which humans can easily answer but can "stump the smart robot" (cf Turing Test!)
- Three questions per image.
- Answers from 10 different people.
- Also asked for answers without seeing the image (22%).

Why does this male have his arms in this position?	balance for balance for balance	angry he's carrying bags hug
Are the clouds high in the sky?	yes yes yes	no no yes

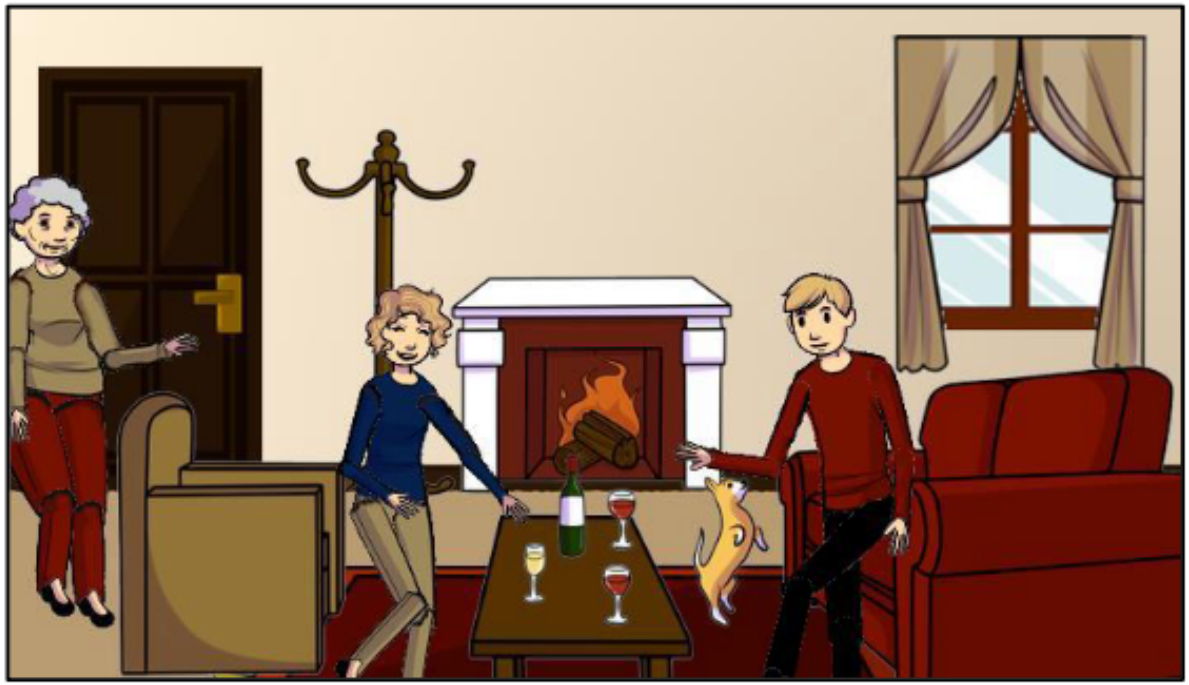
from Agrawal et al (2016)

Which player on the field head-butted the ball?	18 18 player on left	1 in front of goal number 13 number 22
What number is on the girl in black?	18 18 18	1 4 8

from Agrawal et al (2016)

Is this person trying to hit a ball?	yes yes yes	yes yes yes
What is the person hitting the ball with?	frisbie racket round paddle	bat bat racket

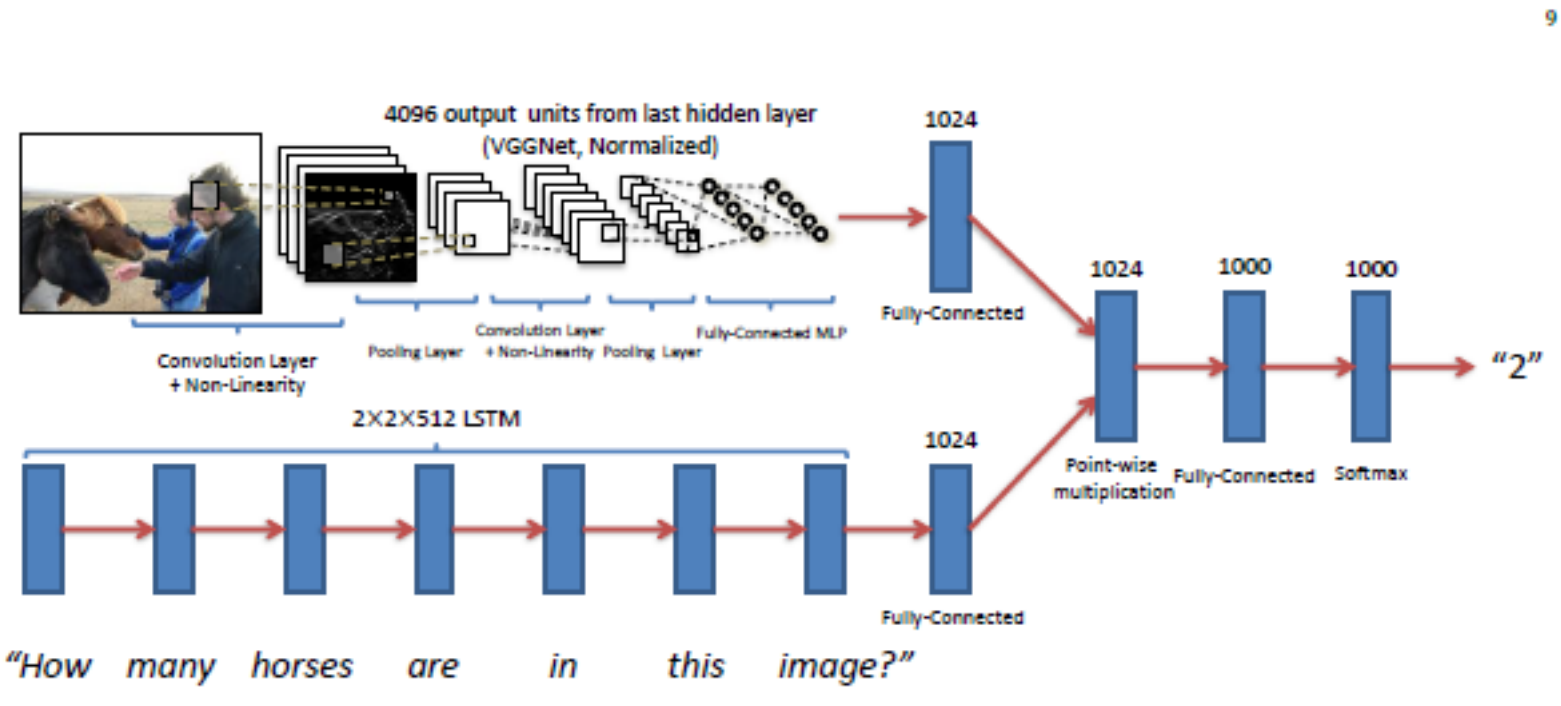
from Agrawal et al (2016)



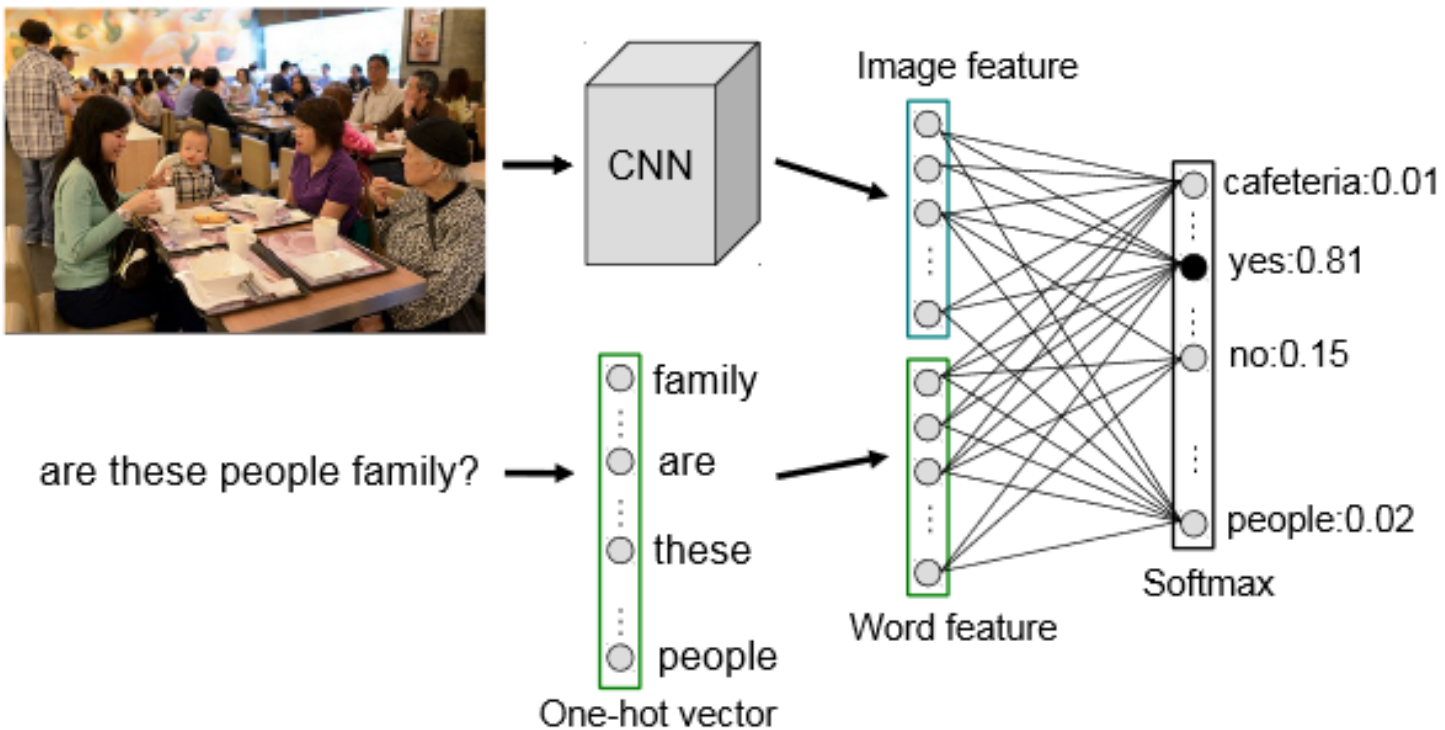
How many glasses are on the table?	3 3 3	2 2 6
What is the woman reaching for?	door handle glass wine	fruit glass remote

from Agrawal et al (2016)

VQA architecture (Agrawal et al, 2016)



Baseline system



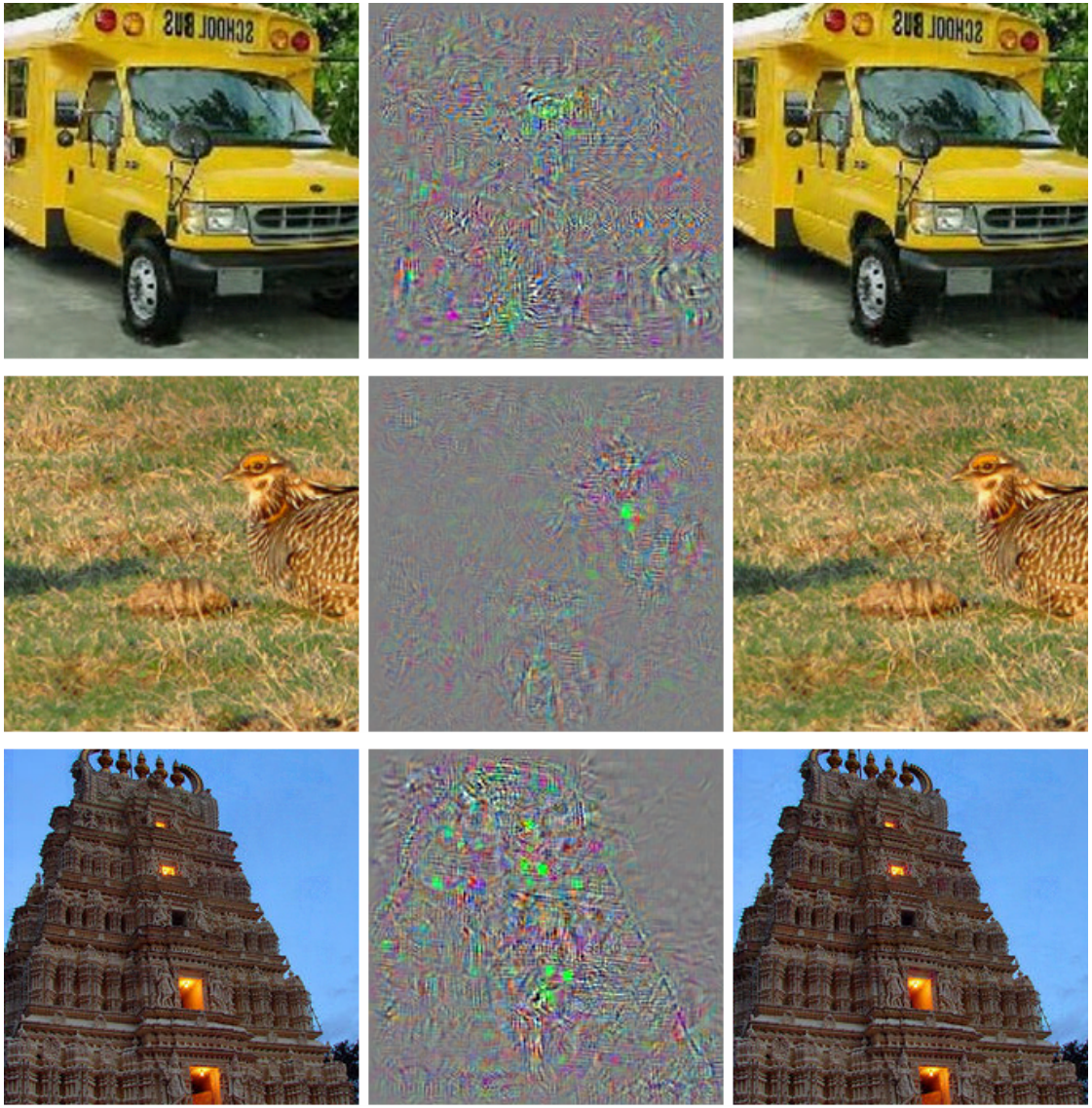
http://visualqa.csail.mit.edu/

Learning commonsense knowledge?

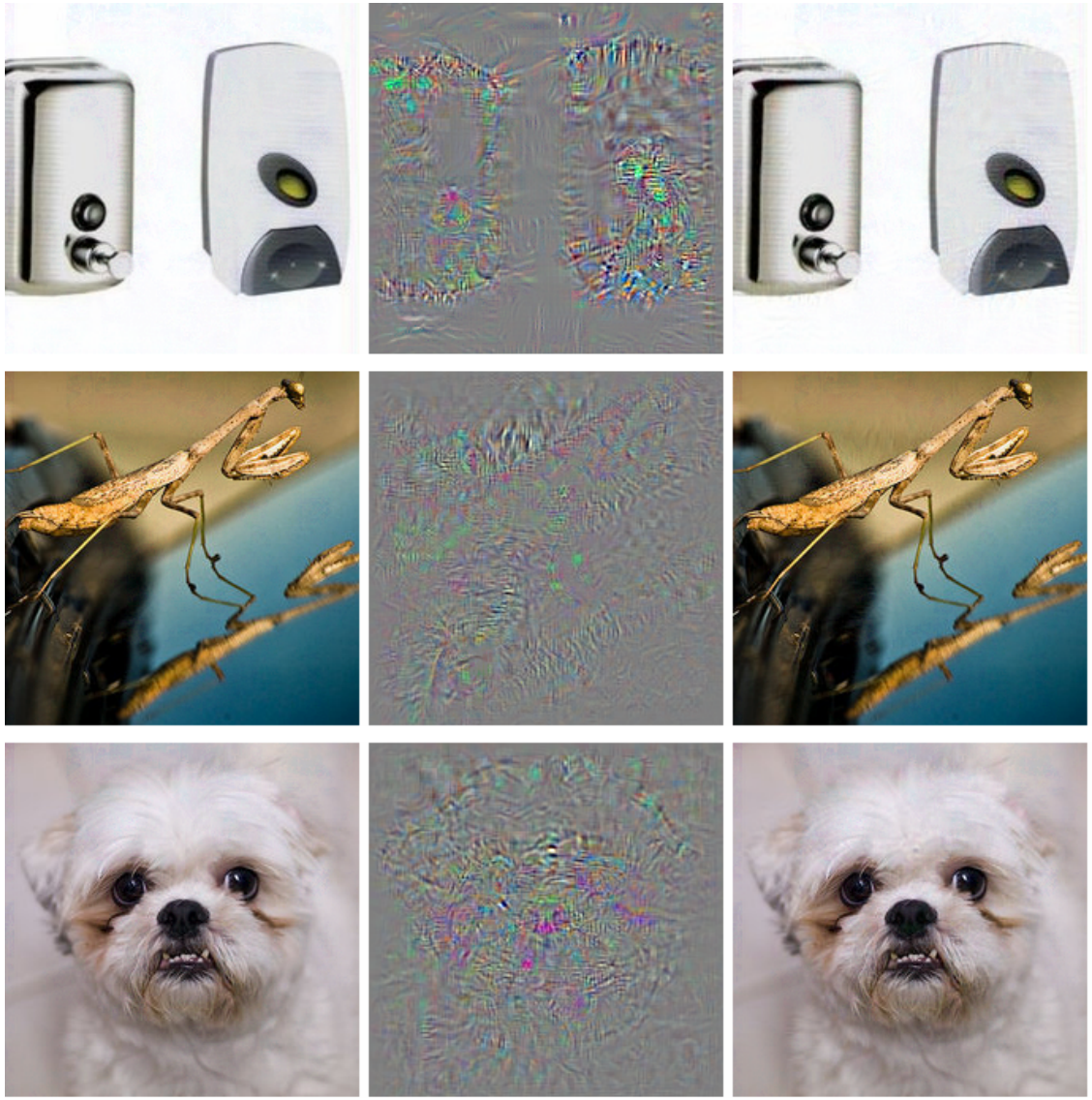
- Zhou et al's baseline system (no hidden layers) performs as well as systems with much more complex architectures (55.7%).
- Correlates input words and visual concepts with the answer.
- Systems are much better than humans at answering without seeing the image (BOW model is at 48%).
- To an extent, the systems are discovering biases in the dataset.
- Systems make errors no human would ever make on unexpected questions: e.g., 'Is there an aardvark?'

Adversarial examples

- For image recognition, images that are correctly recognised are perturbed in a manner imperceptible to a human and are then not recognised.
  https://arxiv.org/pdf/1312.6199.pdf
- Systematically find adversarial examples via low probability 'pockets' (because the space is not smooth): these can't be found efficiently by random sampling around a given example.
- Not clear whether anything directly comparable for NLP: though https://arxiv.org/pdf/1707.07328.pdf for reading comprehension.
- also 'Build it, Break it: the language edition'
  https://bibinlp.umiacs.umd.edu/



https://arxiv.org/pdf/1312.6199.pdf



https://arxiv.org/pdf/1312.6199.pdf

Outline.

Artificial vs biological NNs

- ANNs and BNNs both take input from many neurons and carry out simple processing (e.g., summation), then output to many neurons.
- ANNs are still tiny: biggest c160 billion parameters. Human brain has tens of billions of neurons, each with up to 100,000 synapses.
- Brain connections are much slower than ANNs: chemical transmission across synapse. Bigger size and greater parallelism (more than) makes up for this.
- Neurotransmitters are complex and not well understood: biological neurons are only crudely approximated by on/off firing.

Artificial vs biological NNs (continued)

- Brains grow new synapses and lose old ones: individual brains evolve (Hebbian Learning: “Neurons which fire together wire together”).
- Brains are embodied: processing sensory information, controlling muscles. There is no hard division between these parts of the brain and concepts/reasoning (e.g., experiments with *kick* vs *hit*).
- Brains have evolved over (about) 600 million years (more if we include nerve nets, as in jellyfish).
- Brains are expensive (about 20% of a person’s energy), but much more efficient than ANNs.
- and . . .

Deep learning: positives

- Really important change in state-of-the-art for some applications: e.g., language models for speech.
- Multi-modal experiments are now much more feasible.
- Models are learning structure without hand-crafting of features.
- Structure learned for one task (e.g., prediction) applicable to others with limited training data.
- Lots of toolkits etc
- Huge space of new models, far more research going on in NLP, far more industrial research . . .

Deep Learning: negatives

- Models are made as powerful as possible to the point they are“barely possible to train or use” (`http://www.deeplearningbook.org` 16.7).
- Tuning hyperparameters is a matter of much experimentation.
- Statistical validity of results often questionable.
- Many myths, massive hype and almost no publication of negative results: but there are some NLP tasks where deep learning is not giving much improvement in results.
- Weird results: e.g., ‘33rpm’ normalized to ‘thirty two revolutions per minute’

  `https://arxiv.org/ftp/arxiv/papers/1611/1611.00068.pdf`
- Adversarial examples.