Deep Learning course: Explainability

Thierry Artières

Ecole Centrale Marseille
Data Science Department - LIS laboratory
Machine Learning team - QARMA

November 13, 2019

Plan

Explainabilty

Motivation

- Machine Learning models are black box models
- Needs for trustability comes with the spread of ML and DL in real-life tasks
- Few reasons
  - Verification and validation of models : for sensitive domains
  - Compliance to legislation : defining responsibilities
  - Improvement of the system : when designing it
  - Learning from the models : when superhuman performances
  - End user acceptance : every applications
- Various needs for various domains
  - Defense, Finance
  - Games
  - Automatic driving cars
  - ...
- Few levels of understandability / explainability:
  - Explain a decision on an input
  - Explain the whole model
- Various kind of methods: Agnostic, model-based ...

Explainabilty: approaches

Low level global understanding : Interpreting the neuron's computations

- Interpreting neuron's behaviour : Activation maximization and variants

Local explainability: Explaining Decisions

- Sensitivity analysis
- Layer-wise relevance propagation [Simonyan et al., 2013], ...
- Deconvnets [Zeiler et Fergus 2013]

High level global interpretability at model level

- Getting a simpler model through pruning
  - Regularization : L1, L2, Optimal Cell Damage, Optimal Brain damage...
  - Weights binarization
  - Activation binarization aka model discretization
- Getting a simpler model through distillation
  - Distilling to a simpler NN model
  - Distilling to a more interpretable model (decision tree)

Plan

Feature importance in linear models

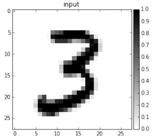
Linear models

- $f(x) = w^T x = \sum_{j=1}^{d} x_j w_j$
- Learning with a regularization term : $C(W) = \sum_{i=1}^{N} l(x^i, y^i, w) + ||w||^2$
- Useless weights go to 0
- Relevance of feature j measured as $|w_j|$

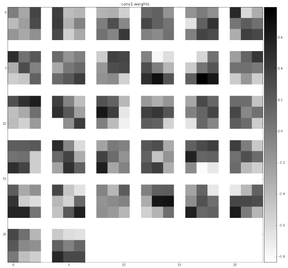
Visualizing filters and activations (primary understanding of NNs)

Mnist (toy) dataset

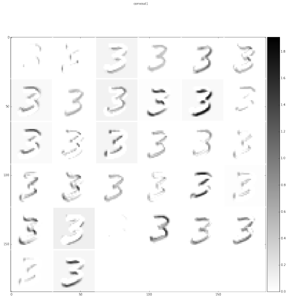
- Low resolution handwritten digit images



Weights of first Convolutional layer (32 maps)



Outputs of first Convolutional layer for above input



A very simple way !

Explore a dataset

- Forward propagate all the dataset in a DNN
- Identify images that most activate a neuron in a dense layer or a neuron in a given convolutional feature map
- Show images or relavant patches of images (crops) that are seen by the neurons

Activation maximization

Principle

- Search for a pattern that maximizes the response of a given neuron
- Example: For a neuron c encoding the posterior class $p(c|x)$ (e.g. through a sotfmax output layer) one may look for :

$$\hat{x} = \arg\max_x \log(p(c|x)) - \lambda \|x\|^2$$

- This may be performed through gradient ascent in the input space
- Good idea but: does not always yield a relevant result

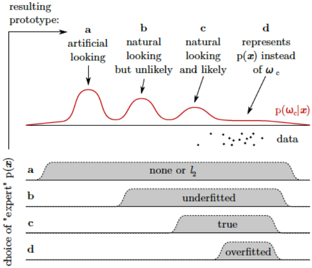
Activation maximization with prior

Main idea

- Use a prior on x, $p(x)$, for getting more reliable results in Activation Maximization methods
- Many choices for the prior, including adversarial learning.
- In this latter case, the gradient ascent is performed on a model chaining the generator AND the DNN one wants to understand

$$\hat{z} = \arg\max_z \log(p(c|g(z))) - \lambda\|z\|^2$$

$$\hat{x} = g(\hat{z})$$



Figures from [Nguyen et al., 2016]

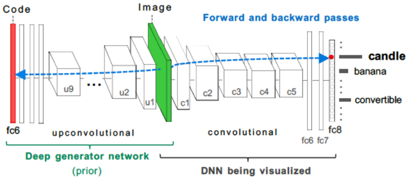
Activation maximization with prior

Main idea

- Use a prior on x, $p(x)$, for getting more reliable results in Activation Maximization methods
- Many choices for the prior, including adversarial learning.
- In this latter case, the gradient ascent is performed on a model chaining the generator AND the DNN one wants to understand

$$\hat{z} = \arg\max_z \log(p(c|g(z))) - \lambda \|z\|^2$$

$$\hat{x} = g(\hat{z})$$



Figures from [Nguyen et al., 2016]

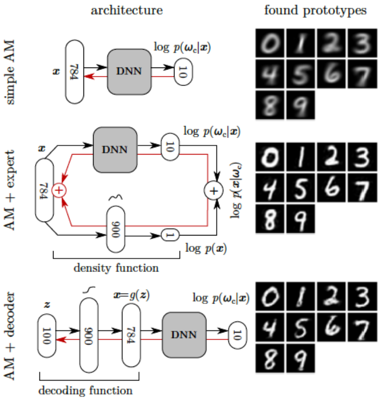
Activation maximization with prior

Main idea

- Use a prior on x, $p(x)$, for getting more reliable results in Activation Maximization methods
- Many choices for the prior, including adversarial learning.
- In this latter case, the gradient ascent is performed on a model chaining the generator AND the DNN one wants to understand

$$\hat{z} = \arg\max_{z} \log(p(c|g(z))) - \lambda \|z\|^2$$

$$\hat{x} = g(\hat{z})$$



Figures from [Nguyen et al., 2016]

Plan

Feature importance in nonlinear models

Linear vs nonlinear models

- Linear models $\rightarrow$ one weight / feature: Minimal interpretability
- Nonlinear models $\rightarrow$ take into account interdependencies between features $\rightarrow$ much more difficult to disentangle the relevance of all the features

Current methods: Gradient etc

- Popular measure: Gradient of class output wrt input features: $|\frac{\partial y^c}{\partial x_j}|$
- Other derived measures

Sensitivity analysis

Goal

- Identify the input features that are responsible for (that explain) the output
- Define scores such as

$$R_i(x) = \left(\frac{\partial f(x)}{\partial x_i}\right)^2$$

- Easy to implement (requires gradient computation)
- Do not explain $f(x)$ but its variation

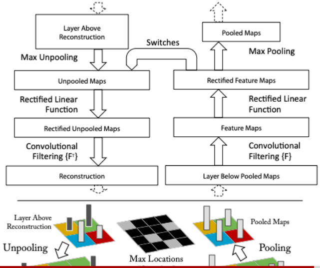
$$\sum_i R_i(x) = \|\nabla f(x)\|^2$$

- Makes the method answer : what makes the input classified in the predicted class rather than in another class ?
- Does not answer: What makes the classifier predict the given class for this input ?

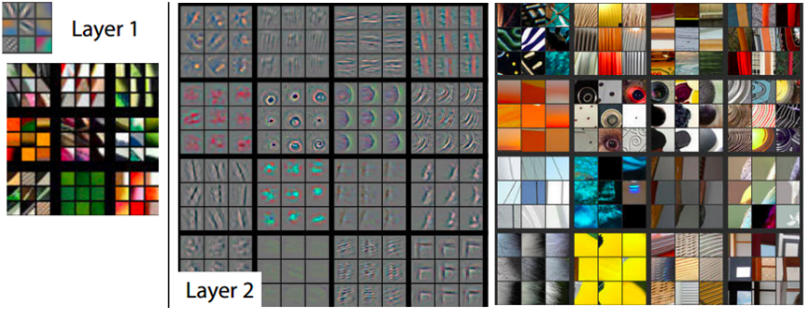
Deconvnet

Principle

- The deconvolutional network ('deconvnet') is an approach for visualizing concepts learned by neurons in higher layers of a CNN
- Given a high-level feature map, the 'deconvnet' inverts the data flow of a CNN, going from neuron activations in the given layer down to an image.
- Typically, a single neuron is left non-zero in the high level feature map.
- The resulting reconstructed image shows the part of the input image that is most strongly activating this neuron



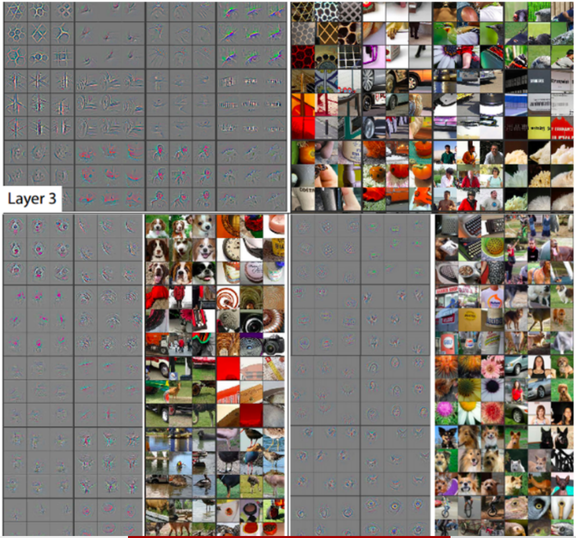
Visualizing filters and activations



Visualizations of Layer 1 and 2. Each layer illustrates 2 pictures, one which shows the filters themselves and one that shows what part of the image are most strongly activated by the given filter. For example, in the space labled Layer 2, we have representations of the 16 different filters (on the left)

[From Zeiler et Fergus]

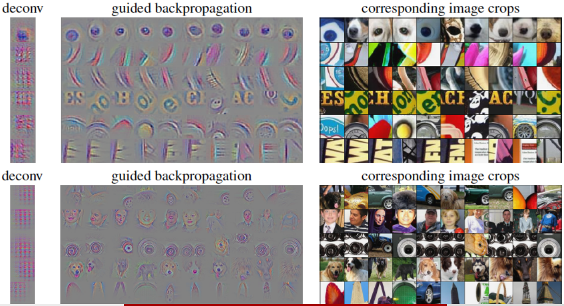
Visualizing filters and activations



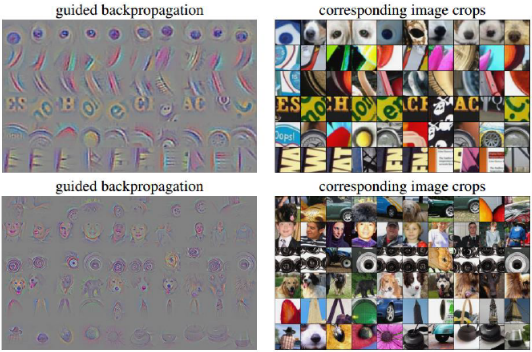
Guided Backpropagation

Principle

- Idea: neurons act like detectors of particular image features
- We are only interested in what image features the neuron detects, not in what kind of stuff it doesn't detect
- So when propagating the gradient, we set all the negative gradients to 0
- We don't care if a pixel "suppresses" a neuron somewhere along the part to our neuron

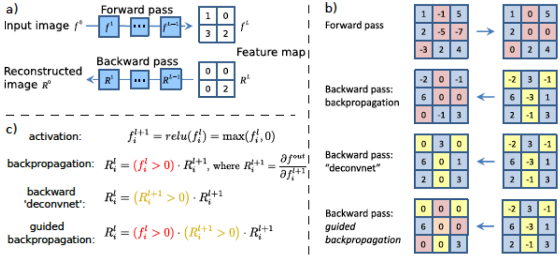


Filters of deep NNs



Springerberg et al, Striving for Simplicity: The All Convolutional Net (ICLR 2015 workshops)

DeconvNet and Guided Backpropagation



Relevance Propopagation

Principle

- Explain the output as the sum of the contribution of all input's features (e.g. pixels in images)

$$f(x) = \sum_i R_i(x)$$

- Multiple rules to compute $R_i(x)$
- Basic idea: redistribute relevance from layer $l+1$ to layer l from the output layer back to the input layer
- For instance:

$$R_j^l = \sum_k \frac{x_j w_{jk}}{\sum_j x_j w_{jk} + \epsilon} R_k^{l+1}$$

- Few variants rule

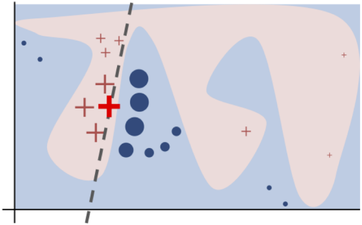
Explaining a decision by sampling around an input [Ribeiro et al., KDD 2016]

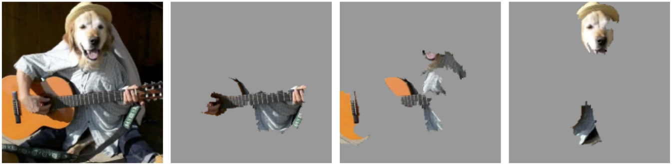
Main idea

- Decision boundary is locally linear (and one may then use standard techniques for linear models)

Method for explaining the decision on input x

- Sample points around a particular input x
- Fit a linear model





(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Plan

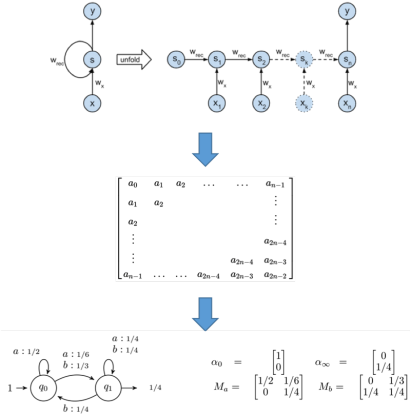
Explaining a model by approximating it

Main idea

- Distillation idea from [Hinton]: Learn a simple model from a large one by using its outputs as targets
- More generally distillation means learning a simpler model to reproduce tje bahavoiour of a complex model
  - May be used with Decision trees
  - Core idea: might be more efficient to learn to behave like a complex model than learning the task from scratch with the simple model

Explaining a model by approximating it

Example on RNNs [Goudian et al., 2018]



Plan

Need for reasoning

Main idea

- Based on a query formulate a goal (information to get)
- Loop
  - Recover the information in the memory that matches the current query
  - Based on gained information and on the original query, formulate another query
- Take a decision based on the accumulated information

Joe went to the kitchen. Fred went to the kitchen. Joe picked up the milk. Joe travelled to the office. Joe left the milk. Joe went to the bathroom. Where is the milk now? A: office Where is Joe? A: bathroom Where was Joe before the office? A: kitchen

Attention mechanism

Main idea

- Most relevant element to a query?
  - Given number of memory elements u_i
  - For a (current) query q
  - Assuming queries and memory elements live un the same space
- The most relevant memory element to query q is the most similar one

$$u* = argmax_i \langle u_i, q \rangle$$

- Use a smooth **argmaximum**

$$s_i = argmax_i \langle u_i, q \rangle$$
$$\alpha_i = softmax(s_i)$$
$$u* = \sum_i \alpha_i u_i$$

Attention mechanism

Implementation in Keras

```
inputs = Input(shape=(input_dims,))
attention_probs = Dense(input_dims, activation='softmax', name='attention_probs')(inputs)
attention_mul = merge([inputs, attention_probs], output_shape=32, name='attention_mul', mode='mul')
```

Simple reasoning and baby stories

Memory Networks [Weston and al., 2015]

- Include a long-term memory that can be read and written to with the goal of using it for prediction: kind of knowledge base
- More straightforward use of the memory than in RNNs
- Ability to deal with complex question answering

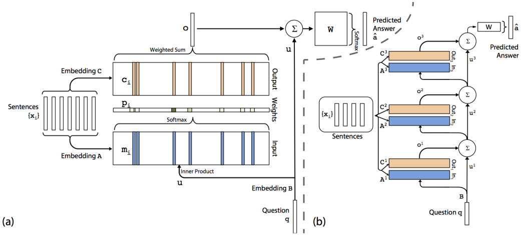


Figure 1: (a): A single layer version of our model. (b): A three layer version of our model. In practice, we can constrain several of the embedding matrices to be the same (see Section 2.2).

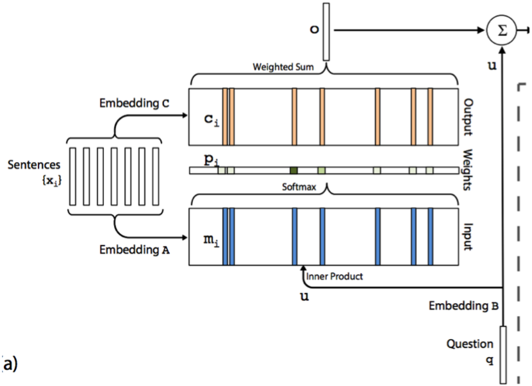
End to End Memory Networks [Sukhbaatar and al., 2015]

Joe went to the kitchen. Fred went to the kitchen. Joe picked up the milk.
Joe travelled to the office. Joe left the milk. Joe went to the bathroom.
Where is the milk now? A: office
Where is Joe? A: bathroom
Where was Joe before the office? A: kitchen

Simple reasoning and baby stories

Memory Networks [Weston and al., 2015]

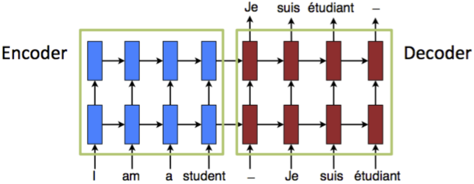
- Zoom on a single hop



Attention mechanisms

Machine translation

- Instead of standard Seq2Seq models
- One may want to focus on one part of input sequence for producing one output word
- Attention = (fuzzy) focus on the input
- Same kind of ideas for automatic captioning

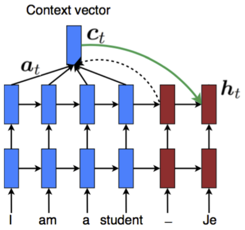


[Bahdanau and al., 2015]

Attention mechanisms

Machine translation

- Instead of standard Seq2Seq models
- One may want to focus on one part of input sequence for producing one output word
- Attention = (fuzzy) focus on the input
- Same kind of ideas for automatic captioning



[Bahdanau and al., 2015]

Attention mechanisms

Machine translation

- Instead of standard Seq2Seq models
- One may want to focus on one part of input sequence for producing one output word
- Attention = (fuzzy) focus on the input
- Same kind of ideas for automatic captioning

A woman is throwing a frisbee in a park. A dog is standing on a hardwood floor. A stop sign is on a road with a mountain in the background.

[Xu et al., 2016]

Attention for translation

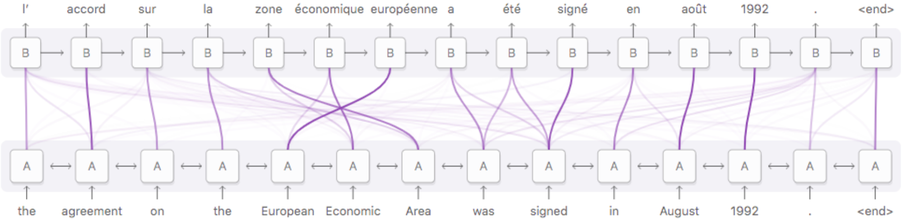


Diagram derived from Fig. 3 of Bahdanau, *et al.* 2014

Attention for speech recognition

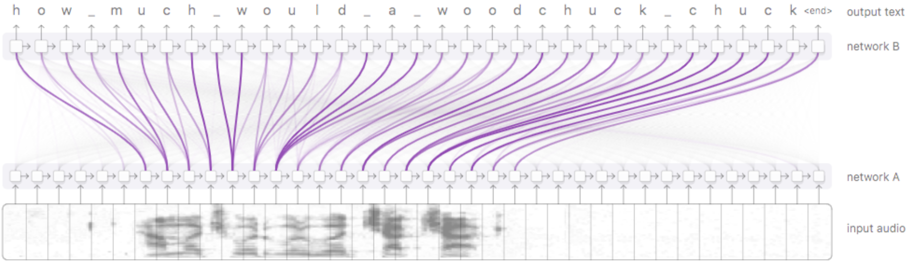
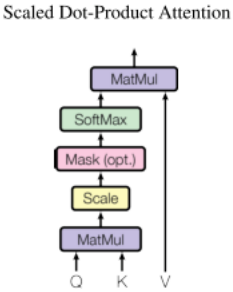


Figure derived from Chan, *et al.* 2015

Self Attention

Main Idea

- Compute new representations of inputs elements in a sequence based on the whole sequence
- $\Rightarrow$ representations of elements in context



- Propagation in the attention layer

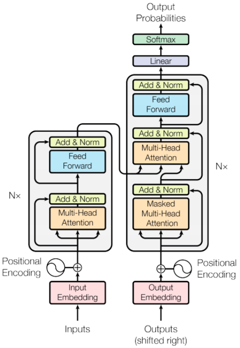
$$Attention(Q, K, V) = softmax\left(\frac{Qk^T}{\sqrt{d}}\right) V$$

- with:
  - Q: queries (with dimension d)
  - K: Keys
  - V: Values
- where Q, K, V could be all equal to the inputs
- But there are transformed (by product with weight matrices) of the inputs

Self Attention

Main Idea

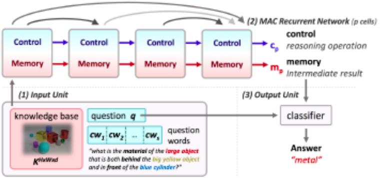
- Compute new representations of inputs elements in a sequence based on the whole sequence
- $\Rightarrow$ representations of elements in context



General reasoning

More complex reasonning tasks

- Requires few steps of question answering like queries



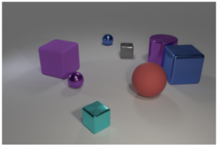


Figure 1: A sample image from the CLEVR dataset, with a question: "There is a purple cube behind a metal object left to a large ball; what material is it?"