Neural Networks for Machine Learning

Lecture 15c
Deep autoencoders for document retrieval and visualization

Geoffrey Hinton
Nitish Srivastava,
Kevin Swersky
Tijmen Tieleman
Abdel-rahman Mohamed
How to find documents that are similar to a query document

• Convert each document into a “bag of words”.
  – This is a vector of word counts ignoring order.
  – Ignore stop words (like “the” or “over”)
• We could compare the word counts of the query document and millions of other documents but this is too slow.
  – So we reduce each query vector to a much smaller vector that still contains most of the information about the content of the document.
How to compress the count vector

- We train the neural network to reproduce its input vector as its output.
- This forces it to compress as much information as possible into the 10 numbers in the central bottleneck.
- These 10 numbers are then a good way to compare documents.
The non-linearity used for reconstructing bags of words

• Divide the counts in a bag of words vector by \( N \), where \( N \) is the total number of non-stop words in the document.
  – The resulting probability vector gives the probability of getting a particular word if we pick a non-stop word at random from the document.

• At the output of the autoencoder, we use a softmax.
  – The probability vector defines the desired outputs of the softmax.

• When we train the first RBM in the stack we use the same trick.
  – We treat the word counts as probabilities, but we make the visible to hidden weights \( N \) times bigger than the hidden to visible because we have \( N \) observations from the probability distribution.
Performance of the autoencoder at document retrieval

• Train on bags of 2000 words for 400,000 training cases of business documents.
  – First train a stack of RBM’s. Then fine-tune with backprop.
• Test on a separate 400,000 documents.
  – Pick one test document as a query. Rank order all the other test documents by using the cosine of the angle between codes.
  – Repeat this using each of the 400,000 test documents as the query (requires 0.16 trillion comparisons).
• Plot the number of retrieved documents against the proportion that are in the same hand-labeled class as the query document. Compare with LSA (a version of PCA).
Retrieval performance on 400,000 Reuters business news stories

![Graph showing accuracy vs. number of retrieved documents for Autoencoder-10D, LSA-50D, and LSA-10D.]
First compress all documents to 2 numbers using PCA on \( \log(1+\text{count}) \). Then use different colors for different categories.
First compress all documents to 2 numbers using deep auto. Then use different colors for different document categories.
Lecture 15d
Semantic hashing

Geoffrey Hinton
Nitish Srivastava,
Kevin Swersky
Tijmen Tieleman
Abdel-rahman Mohamed
Finding binary codes for documents

- Train an auto-encoder using 30 logistic units for the code layer.
- During the fine-tuning stage, add noise to the inputs to the code units.
  - The noise forces their activities to become bimodal in order to resist the effects of the noise.
  - Then we simply threshold the activities of the 30 code units to get a binary code.
- Krizhevsky discovered later that its easier to just use binary stochastic units in the code layer during training.
Using a deep autoencoder as a hash-function for finding approximate matches

hash function

supermarket search

Document

Memory

Semantically Similar Documents