AMMI Deep Learning DIY:

Day 8

Timothée Lacroix, Pierre Stock, Alexandre Sablayrolles

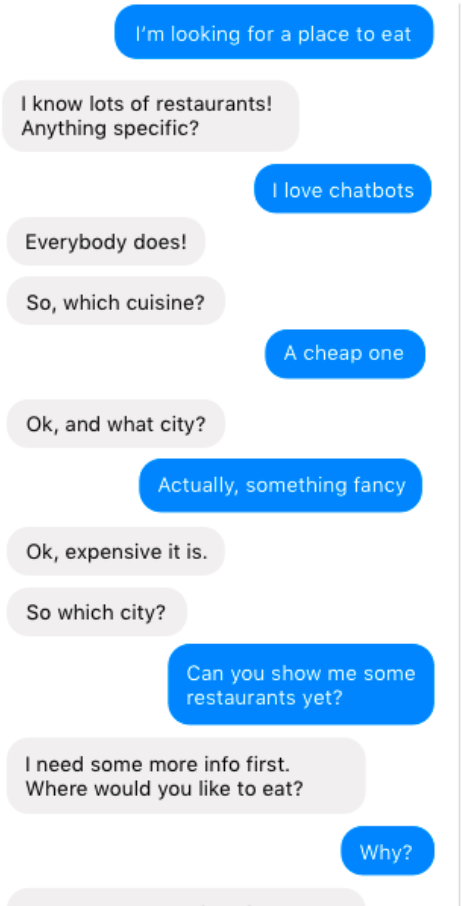
Natural Language Processing (NLP)

Some applications of NLP

- Why are we interested in Natural Language Processing ?

Some applications of NLP

- Why are we interested in Natural Language Processing ?
- Conversations & chatbots



Chitchat: Making small-talk or asking questions unrelated to the task

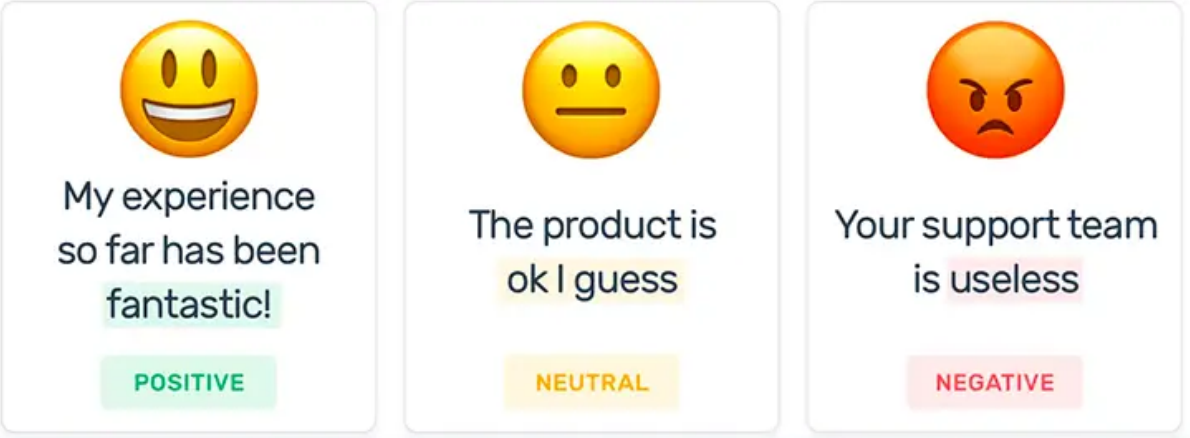
Correction: User has changed their mind about something

Broad context: Questions referring to the broader context of conversation, such as the status of the current task

Narrow context: Questions referring to immediate context, e.g. the most recent system utterance

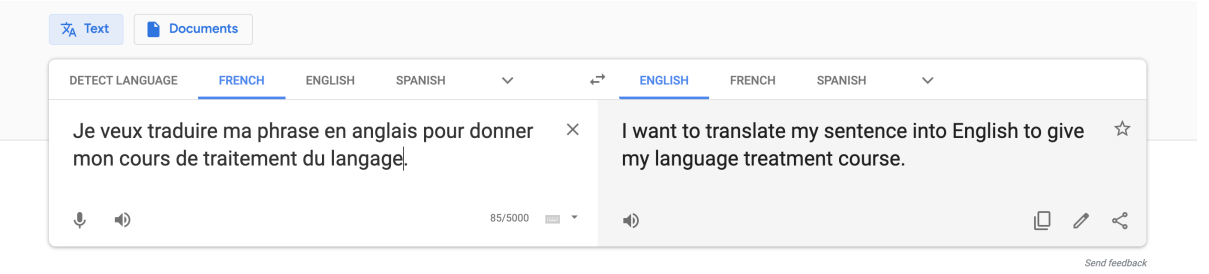
Some applications of NLP

- Why are we interested in Natural Language Processing ?
- Text classification / sentiment analysis



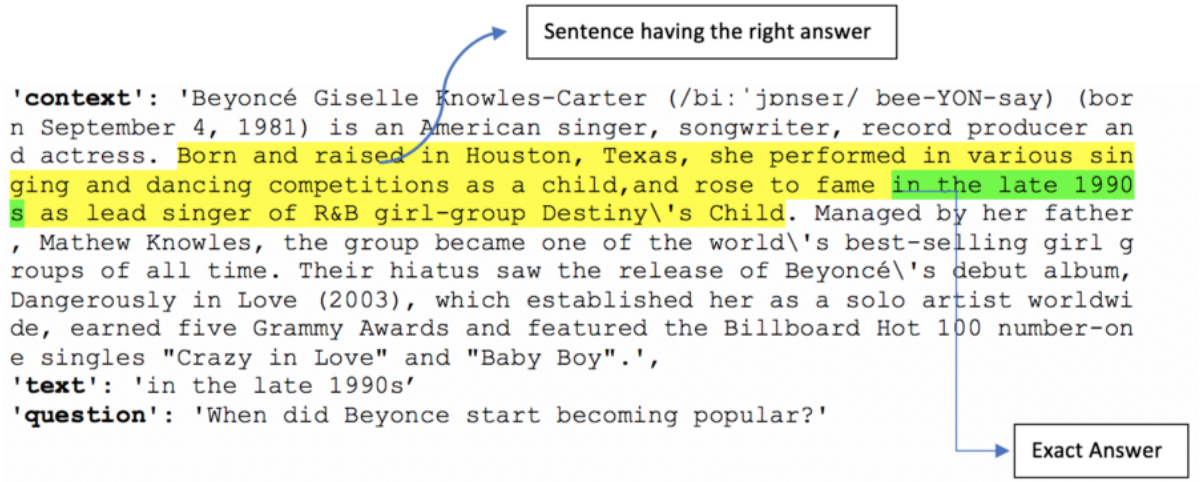
Some applications of NLP

- Why are we interested in Natural Language Processing ?
- Machine translation



Some applications of NLP

- Why are we interested in Natural Language Processing ?
- Question answering



Some applications of NLP

- Why are we interested in Natural Language Processing ?
- +audio: Speech recognition

Some applications of NLP

- Why are we interested in Natural Language Processing ?
- +image: Image captioning

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

Natural Language Processing

1. Word vectors
2. Recurrent models: RNN, LSTM

Word vectors / embeddings

- We want to represent each word by a vector (or embedding)

Word vectors / embeddings

- We want to represent each word by a vector (or embedding)
- Why ?

Word vectors / embeddings

- We want to represent each word by a vector (or embedding)
- Why ?
  - Our machines (linear regression, neural nets) work with vectors (i.e. feature representations)

Word vectors

1. Learning word vectors

Word vectors

1. Learning word vectors
    - word2vec (skip-gram, CBOW)

Word vectors

1. Learning word vectors
    - word2vec (skip-gram, CBOW)
2. Properties of learned word vectors

Word vectors

1. Learning word vectors
    - word2vec (skip-gram, CBOW)
2. Properties of learned word vectors
    - linear relationships, neighbors, bias

Word vectors

1. Learning word vectors
    - word2vec (skip-gram, CBOW)
2. Properties of learned word vectors
    - linear relationships, neighbors, bias
    - visualization

Word vectors

1. Learning word vectors
    - word2vec (skip-gram, CBOW)
2. Properties of learned word vectors
    - linear relationships, neighbors, bias
    - visualization
3. Transferring to downstream applications

Word vectors

1. Learning word vectors
    - word2vec (skip-gram, CBOW)
2. Properties of learned word vectors
    - linear relationships, neighbors, bias
    - visualization
3. Transferring to downstream applications
    - text classification / sentiment analysis

Word vectors

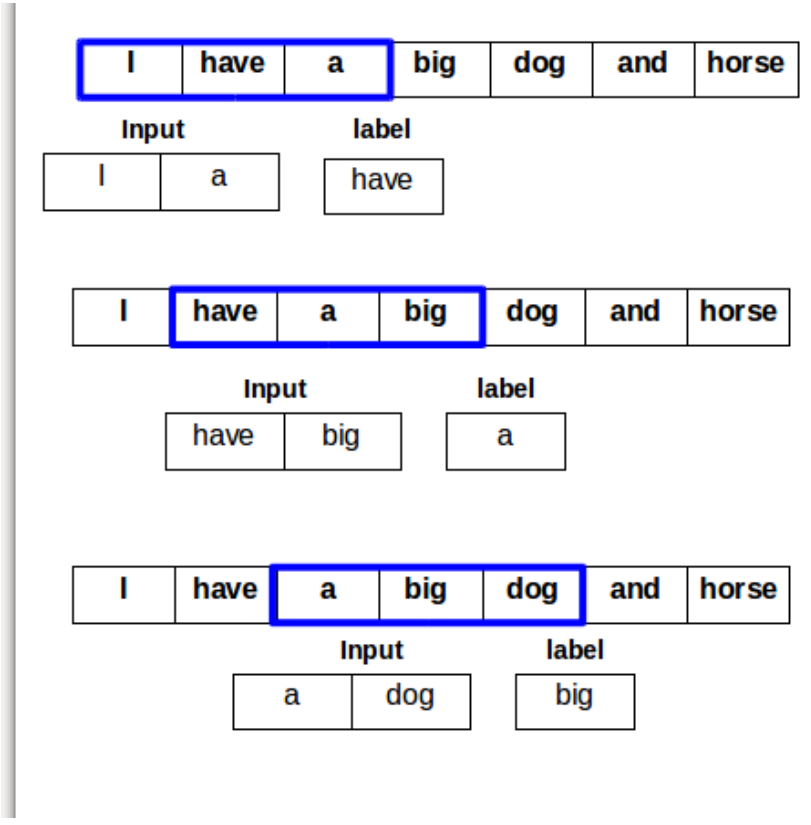
1. Learning word vectors
    - word2vec (skip-gram, CBOW)
2. Properties of learned word vectors
    - linear relationships, neighbors, bias
    - visualization
3. Transferring to downstream applications
    - text classification / sentiment analysis
    - embedding layers

Learning word vectors

- Word vectors can be learned in an unsupervised way

Learning word vectors

- Word vectors can be learned in an unsupervised way
- Given a corpus of text, we predict a word from its context



Learning word vectors

- Word2vec (Mikolov et al. 2013)

Learning word vectors

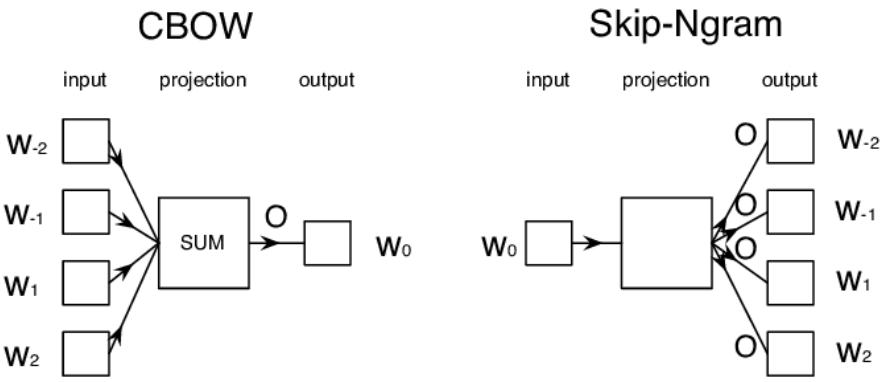
- Word2vec (Mikolov et al. 2013)
  - CBOW: predict a word from its context

Learning word vectors

- Word2vec (Mikolov et al. 2013)
  - CBOW: predict a word from its context
  - Skip-gram: predict the context from the word

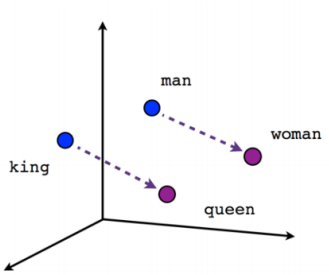
Learning word vectors

- Word2vec (Mikolov et al. 2013)
  - CBOW: predict a word from its context
  - Skip-gram: predict the context from the word

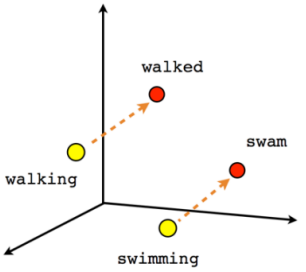


Properties: linear relationships

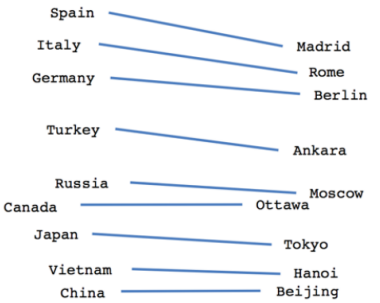
- Linear relationships between words
  - There are "directions" in the learned space
  - ex: country-capital, male-female, etc.



Male-Female



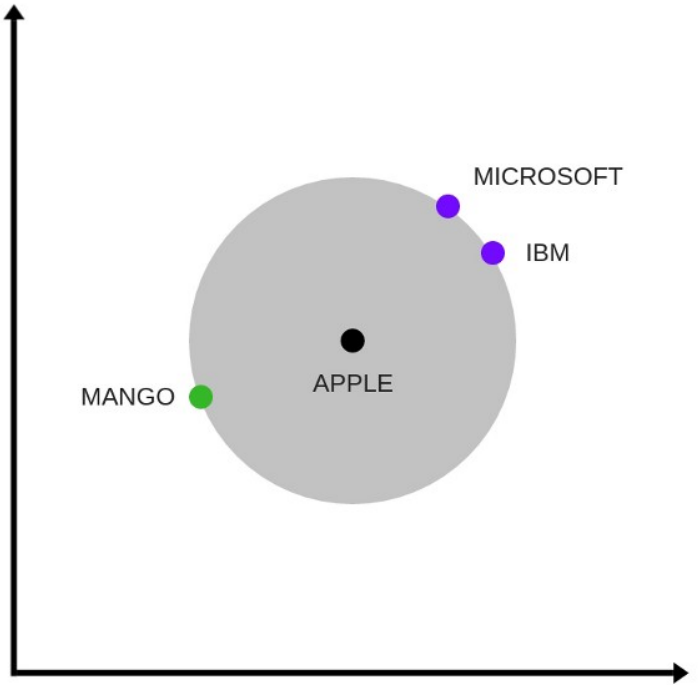
Verb tense



Country-Capital

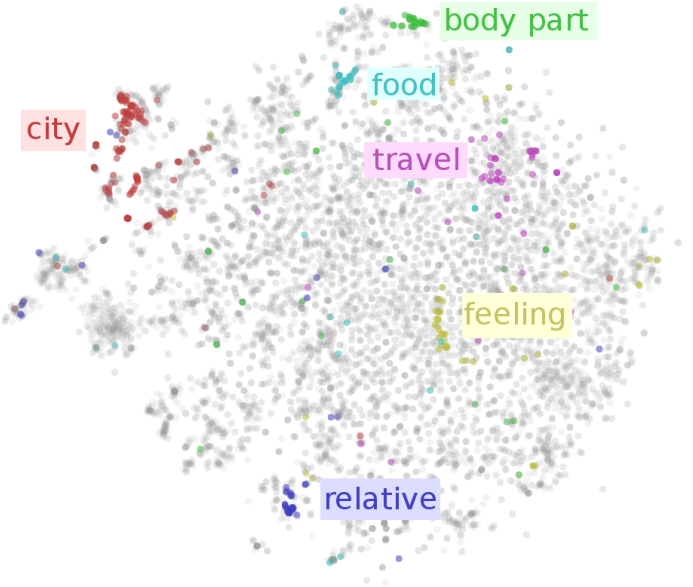
Properties: nearest neighbors

- Nearest neighbors of points are often semantically related



Properties: visualization

- T-SNE plot of word embeddings



Properties: bias!

- Our representation can be biased!

Properties: bias!

- Our representation can be biased!
- This is not something we want, but we have to be aware of it

Properties: bias!

- Our representation can be biased!
- This is not something we want, but we have to be aware of it

$$\text{woman} + \text{computer programmer} - \text{man} \approx \textbf{home maker}$$

Properties: bias!

- Our representation can be biased!
- This is not something we want, but we have to be aware of it

$$\text{woman} + \text{computer programmer} - \text{man} \approx \textbf{home maker}$$

- Why do we have bias ?

Properties: bias!

- Our representation can be biased!
- This is not something we want, but we have to be aware of it

$$\text{woman} + \text{computer programmer} - \text{man} \approx \textbf{home maker}$$

- Why do we have bias ?
  - Our data is biased

Properties: bias!

- Our representation can be biased!
- This is not something we want, but we have to be aware of it

$$\text{woman} + \text{computer programmer} - \text{man} \approx \textbf{home maker}$$

- Why do we have bias ?
  - Our data is biased
  - We did not do anything to correct it

Properties: bias!

- Our representation can be biased!
- This is not something we want, but we have to be aware of it

$$\text{woman} + \text{computer programmer} - \text{man} \approx \textbf{home maker}$$

- Why do we have bias ?
  - Our data is biased
  - We did not do anything to correct it
- Bias can be removed

Properties: bias!

- Our representation can be biased!
- This is not something we want, but we have to be aware of it

$$\text{woman} + \text{computer programmer} - \text{man} \approx \textbf{home maker}$$

- Why do we have bias ?
  - Our data is biased
  - We did not do anything to correct it
- Bias can be removed

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings

Tolga Bolukbasi[1], **Kai-Wei Chang**[2], **James Zou**[2], **Venkatesh Saligrama**[1,2], **Adam Kalai**[2]
[1]Boston University, 8 Saint Mary's Street, Boston, MA
[2]Microsoft Research New England, 1 Memorial Drive, Cambridge, MA
tolgab@bu.edu, kw@kwchang.net, jamesyzou@gmail.com, srv@bu.edu, adam.kalai@microsoft.com

Properties: bias!

- Our representation can be biased!
- This is not something we want, but we have to be aware of it

$$\text{woman} + \text{computer programmer} - \text{man} \approx \textbf{home maker}$$

- Why do we have bias ?
  - Our data is biased
  - We did not do anything to correct it
- Bias can be removed

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings

Tolga Bolukbasi[1], **Kai-Wei Chang**[2], **James Zou**[2], **Venkatesh Saligrama**[1,2], **Adam Kalai**[2]
[1]Boston University, 8 Saint Mary's Street, Boston, MA
[2]Microsoft Research New England, 1 Memorial Drive, Cambridge, MA
tolgab@bu.edu, kw@kwchang.net, jamesyzou@gmail.com, srv@bu.edu, adam.kalai@microsoft.com

- This is still an active research topic

Transfer: linear models

- We can obtain word vectors "for free": we only need text corpora

Transfer: linear models

- We can obtain word vectors "for free": we only need text corpora
  - books

Transfer: linear models

- We can obtain word vectors "for free": we only need text corpora
  - books
  - websites (news, blogs)

Transfer: linear models

- We can obtain word vectors "for free": we only need text corpora
  - books
  - websites (news, blogs)
  - etc.

Transfer: linear models

- We can obtain word vectors "for free": we only need text corpora
  - books
  - websites (news, blogs)
  - etc.
- We can now use these word vectors for text classification

Transfer: linear models

- We can obtain word vectors "for free": we only need text corpora
  - books
  - websites (news, blogs)
  - etc.
- We can now use these word vectors for text classification
- Sentences are represented with the so-called "bag of words"

Transfer: linear models

- We can obtain word vectors "for free": we only need text corpora
  - books
  - websites (news, blogs)
  - etc.
- We can now use these word vectors for text classification
- Sentences are represented with the so-called "bag of words"
  - Each sentence is the average of all its word embeddings

Transfer: linear models

- We can obtain word vectors "for free": we only need text corpora
  - books
  - websites (news, blogs)
  - etc.
- We can now use these word vectors for text classification
- Sentences are represented with the so-called "bag of words"
  - Each sentence is the average of all its word embeddings
  - We train a simple linear model on top of it

Transfer: embedding layer

- Word vectors can be used as the first layer in a neural network
- More on that later

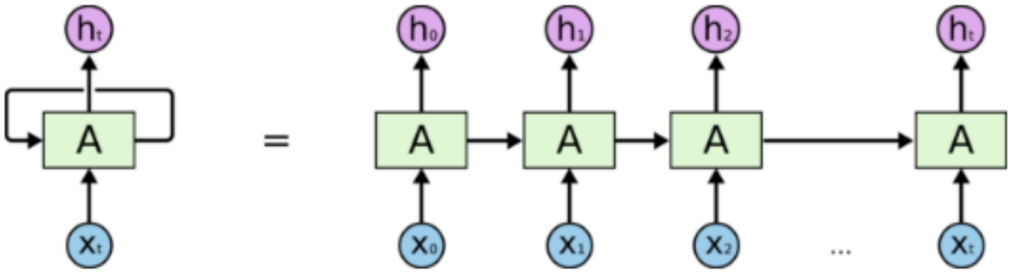
Recurrent neural networks

- Main idea: we predict the next word in a sentence from the current word and the past
- The past is represented by a hidden state $_{-1}$
- The current word is and its embedding is
- We compute the current state by the equation:

$$h_t = \sigma(Ah_{t-1} + Bx_t)$$

- We predict the next word using

Recurrent neural networks



An unrolled recurrent neural network.

Recurrent neural networks

- Also do unsupervised learning!

Recurrent neural networks

- Also do unsupervised learning!
  - We predict the next word with the current word

Recurrent neural networks

- Also do unsupervised learning!
  - We predict the next word with the current word
- There are fancier versions of RNN (LSTM)

Recurrent neural networks

- Also do unsupervised learning!
  - We predict the next word with the current word
- There are fancier versions of RNN (LSTM)
- It was observed that the gradient of RNN blows up

Recurrent neural networks

- Also do unsupervised learning!
  - We predict the next word with the current word
- There are fancier versions of RNN (LSTM)
- It was observed that the gradient of RNN blows up
  - Why ?

Recurrent neural networks

- Also do unsupervised learning!
  - We predict the next word with the current word
- There are fancier versions of RNN (LSTM)
- It was observed that the gradient of RNN blows up
  - Why ?
  - At each time step, we backpropagate the gradient to the previous time step by multiplying it with a matrix

Recurrent neural networks

- Also do unsupervised learning!
  - We predict the next word with the current word
- There are fancier versions of RNN (LSTM)
- It was observed that the gradient of RNN blows up
  - Why ?
  - At each time step, we backpropagate the gradient to the previous time step by multiplying it with a matrix
  - After steps, it can blow up or vanish exponentially

Recurrent neural networks

- Also do unsupervised learning!
  - We predict the next word with the current word
- There are fancier versions of RNN (LSTM)
- It was observed that the gradient of RNN blows up
  - Why ?
  - At each time step, we backpropagate the gradient to the previous time step by multiplying it with a matrix
  - After steps, it can blow up or vanish exponentially
  - This is called the exploding/vanishing gradient problem

Recurrent neural networks

- Also do unsupervised learning!
  - We predict the next word with the current word
- There are fancier versions of RNN (LSTM)
- It was observed that the gradient of RNN blows up
  - Why ?
  - At each time step, we backpropagate the gradient to the previous time step by multiplying it with a matrix
  - After steps, it can blow up or vanish exponentially
  - This is called the exploding/vanishing gradient problem
  - It has a simple solution: gradient clipping

Recurrent neural networks

- Also do unsupervised learning!
  - We predict the next word with the current word
- There are fancier versions of RNN (LSTM)
- It was observed that the gradient of RNN blows up
  - Why ?
  - At each time step, we backpropagate the gradient to the previous time step by multiplying it with a matrix
  - After steps, it can blow up or vanish exponentially
  - This is called the exploding/vanishing gradient problem
  - It has a simple solution: gradient clipping
  - If the norm of the gradient is higher than some threshold, we rescale it