Neural Networks in MT: Past, Present and Future

Holger Schwenk

September 12, 2016

Plan of the Talk

- Machine translation: more than 60 years of research
- Deep neural networks: why, when and how
- The path from neural language models
  to fully neural machine translation
- Global Joint training
  and sentence representations
- Conclusion and perspectives

History of Machine Translation

- Machine Translation is one of the oldest domains in Computer Science

Sentences in Russian are punched into standard cards for feeding into the electronic data processing machine for translation into English

History of Machine Translation

- Machine Translation is one of the oldest domains in Computer Science

Sentences in Russian are punched into standard cards for feeding into the electronic data processing machine for translation into English

- First system by IBM in 1954 (Georgetown): translation of Russian into English of 60 sentences

History of Machine Translation

- Machine Translation is one of the oldest domains in Computer Science

Sentences in Russian are punched into standard cards for feeding into the electronic data processing machine for translation into English

- First system by IBM in 1954 (Georgetown): translation of Russian into English of 60 sentences

$\Rightarrow$ Great enthusiasm and multiple research projects

Approaches in Machine Translation

1954	IBM Georgetown, rule-based system Ru/En
1966	ALPAC report stopped funding and research
1968	Creation of the company SYSTRAN
1981	Meteo system (used until 2001)
1993	Statistical MT: IBM1-5 word-based models
2003	Phrase-based MT
2005	Moses platform: fostered widespread research
2005	Hierarchical systems
2005–16	Many incremental improvements of PBSMT
2006	First use of neural networks in MT (LM rescoring)
2014	First fully neural MT
2016	NMT outperforms PBSMT in WMT evaluation

Incremental Improvements of Phrase-Based SMT

- MERT
- Lexicalized reordering models
- Language models trained on huge amounts of data
- Factored translation models
- Decoding: stack or cube pruning, MBR, ...
- Domain adaptation
- Data selection and instance weighting
- Sparse features + PRO/MIRA

⋮

⇒ State-of-the-art phrase-based systems:

- combination of **many individually optimized modules**
- **many heuristics** and "hacks" to resolve observed errors
- **increasingly complicated to train**

Neural Networks in Machine Translation

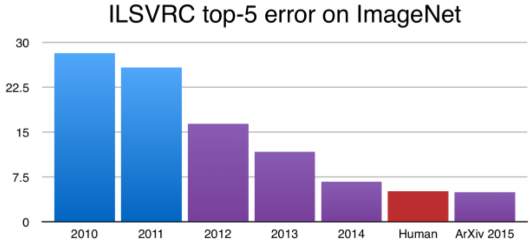
- Neural network language model [Bengio et al, NIST'02]
- CSLM rescoring for SMT [Schwenk et al, ACL'06]
- Neural tuple-based MT [Schwenk et al, EMNLP'07]
- First neural translation models:
  [Schwenk Coling'12, Le et al, NAACL'12, Auli et al EMNIP'13]
- Neural network joint models [Devlin et al, ACL'14]
- Sequence-to-sequence models [Kalchbrenner et al EMNLP'13, Cho et al, EMNLP'14, Sutskever et al NIPS'14]
- Attention mechanism [Bahdanau et al ICLR'15]
- Neural models outperform PBSMT in many language pairs at WMT'16
- Deep NMT systems [Baidu TACL'16]

$\Rightarrow$ Since 2014, the community definitely switched to NN

Deep Neural Networks in Computer Vision

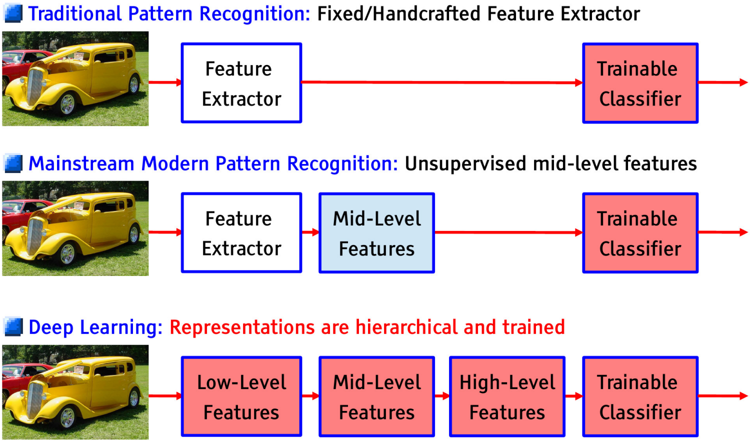
Image net challenge

- Train: 1.2M images with 1000 classes, test: 200k images
- Evolution of error rates:



- The classification error decreased from 28 to less than 4% and reaches today human performance
- Deep neural networks are used since 2012

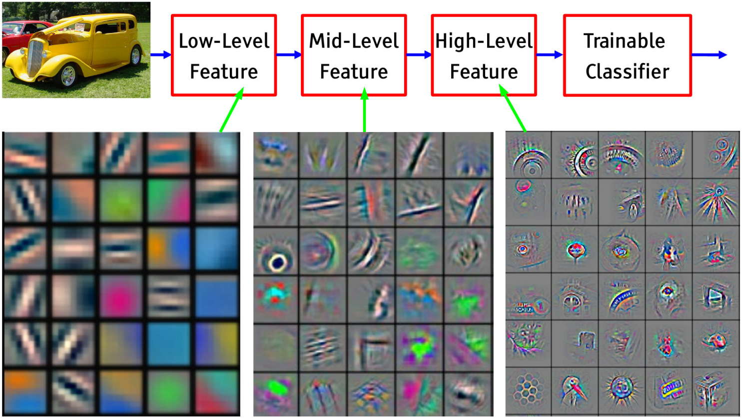
Learning Hierarchical Representations



(figure from Y. Le Cun)

Learning Hierarchical Representations

- It's deep if it has more than one stage of non-linear feature transformation



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Learning Hierarchical Representations

Image recognition

- pixel $\rightarrow$ edge $\rightarrow$ tecton $\rightarrow$ motif $\rightarrow$ part $\rightarrow$ object

Learning Hierarchical Representations

Image recognition

- pixel → edge → tecton → motif → part → object

Text processing

- char → word → word group → clause → sentence → story

Learning Hierarchical Representations

Image recognition

- pixel → edge → tecton → motif → part → object

Text processing

- char → word → word group → clause → sentence → story

Speech recognition

- wave → spectral band → sound → phone → word → sentence

Learning Hierarchical Representations

Image recognition

- pixel → edge → tecton → motif → part → object

Text processing

- char → word → word group → clause → sentence → story

Speech recognition

- wave → spectral band → sound → phone → word → sentence

The intermediate features do not necessarily correspond to a well defined entity for humans !

Network Architectures : ConvNet

Background

- In principle, any problem can be solved with a fully connected (deep) neural network
- However, it is very hard to learn the best solution due to the huge search space

$\Rightarrow$ Constrain the network architecture to be problem-specific

Network Architectures : ConvNet

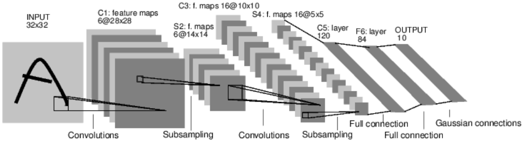
Background

- In principle, any problem can be solved with a fully connected (deep) neural network
- However, it is very hard to learn the best solution due to the huge search space

⇒ Constrain the network architecture to be problem-specific

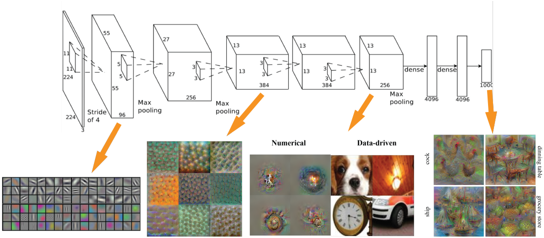
Convolutional networks

- Several layers of small feature detectors and pooling



Network Architectures : ConvNet

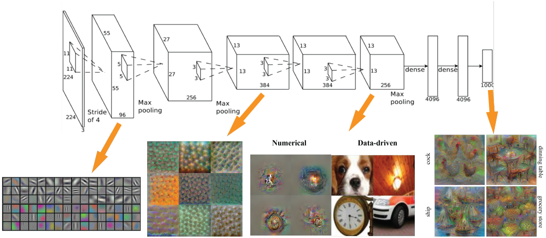
Improved version: GoogleNet



Conv 1: Edge+Blob   Conv 3: Texture   Conv 5: Object Parts   Fc8: Object Classes

Network Architectures : ConvNet

Improved version: GoogleNet



Conv 1: Edge+Blob Conv 3: Texture Conv 5: Object Parts Fc8: Object Classes

Revolution of Depth

- ResNet with 150 layers (or even up to 1000 !)

What about Deep Neural Networks in NLP ?

- Operate on a low level representation of the data
  - Vision: pixels
  - NLP: what is the fundamental unit - words or characters ?
    discrete units !
- Use very deep architectures to learn hierachical representations of the data
  - Vision: feature detectors of increasing abstraction
  - NLP: how to structure the input ?
    n-grams, syntactical or semantic graphs, ... ?
- Structure the network to adapt it to the problem
  - Vision: ConvNets implement learnable feature detectors
  - NLP: Recurrent NN (LSTM, GRU) are very popular
    ConvNets can also be used
- Trained end-to-end
  - Vision: classification problems are well-defined
  - NLP: sentence generation is often ambigious, without unique solution

Natural Language Processing

Handling words

- Detect relationships between words
- Associate categories to a (sequences of) words
- Estimate probability distributions over (sequence) of words
- Generate sentences

⋮

Natural Language Processing

Handling words

- Detect relationships between words
- Associate categories to a (sequences of) words
- Estimate probability distributions over (sequence) of words
- Generate sentences

⋮

Old technique

- Define a vocabulary of V known words
- Represent each word by an integer index
- 1-out-of-N encoding, binary vector

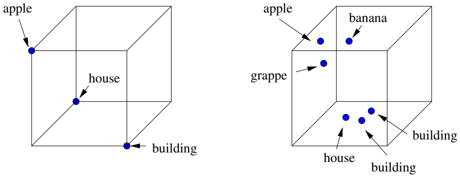
$\Rightarrow$ There is no relation between the words

$\Rightarrow$ All the words are equally close or far

Word Embeddings

Idea

- Associate an arbitrary vector $x_i \in R^E$ to each word
- Learn these embeddings in a way that *similar* words are nearby in that space



- The notion of similarity may depend on the application (LM, MT, dialog, ...)

$\Rightarrow$ There is probably no *"universal word embedding"*

Word Embeddings

How to learn word embeddings ?

- There are many techniques to learn word embeddings
- Some well known / frequently used techniques:
    - Neural language model [Bengio et al, 2001]
    - Word2Vec [Mikolov et al]
- It is usually best to learn the embeddings jointly with the task
- A good initialization may speed up training
  or help to cover unseen words

Continuous Space LM

Theoretical drawbacks of back-off LM:

- Words are represented in a high-dimensional discrete space
- Probability distributions are not smooth functions
- Any change of the word indices can result in an arbitrary change of LM probability

$\Rightarrow$ True generalization is difficult to obtain

Continuous Space LM

Theoretical drawbacks of back-off LM:

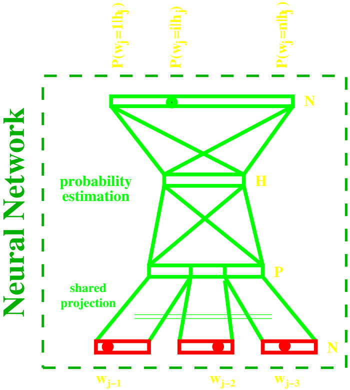
- Words are represented in a high-dimensional discrete space
- Probability distributions are not smooth functions
- Any change of the word indices can result in an arbitrary change of LM probability

$\Rightarrow$ True generalization is difficult to obtain

Main idea [Y. Bengio, NIPS'01]:

- Project word indices onto a continuous space and use a probability estimator operating on this space
- Probability functions are smooth functions and better generalization can be expected

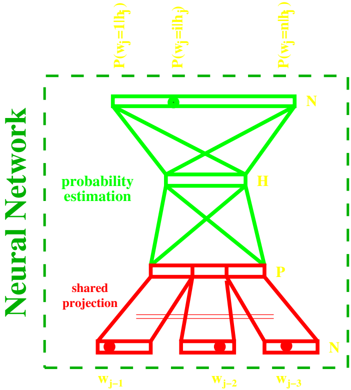
CSLM - Probability Calculation



$$h_j = w_{j-n+1}, \ldots, w_{j-2}, w_{j-1}$$

- Inputs = indices of the $n-1$ previous words

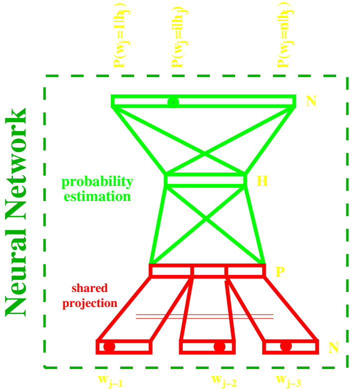
CSLM - Probability Calculation



$$h_j = w_{j-n+1}, \ldots, w_{j-2}, w_{j-1}$$

- Projection onto continuous space
- Inputs = indices of the $n-1$ previous words

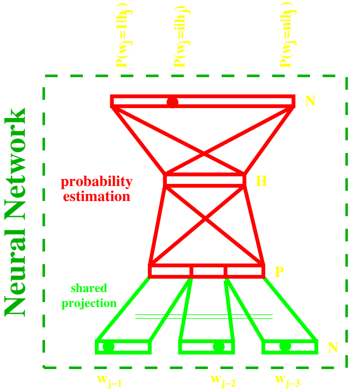
CSLM - Probability Calculation



$$h_j = w_{j-n+1}, \dots, w_{j-2}, w_{j-1}$$

- Context h_j = sequence of $n{-}1$ points in this space
- Word = point in the P dimensional space
- Projection onto continuous space
- Inputs = indices of the $n{-}1$ previous words

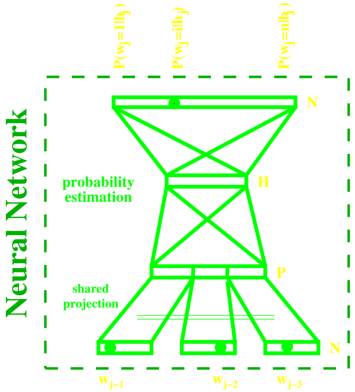
CSLM - Probability Calculation



$h_j = w_{j-n+1}, ..., w_{j-2}, w_{j-1}$

- Outputs = LM posterior probabilities of all words: $P(w_j = i|h_j) \quad \forall i \in [1, N]$
- Context h_j = sequence of $n-1$ points in this space
- Word = point in the P dimensional space
- Projection onto continuous space
- Inputs = indices of the $n-1$ previous words

CSLM - Training

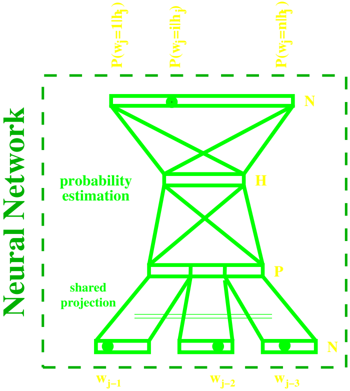


- Backprop training, cross-entropy error

$$E = \sum_{i=1}^{N} d_i \log p_i$$

$+$ weight decay

CSLM - Training



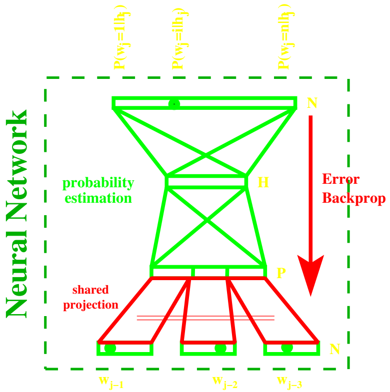
- Backprop training, cross-entropy error

$$E = \sum_{i=1}^{N} d_i \; log \; p_i$$

$+$ weight decay

$\Rightarrow$ NN minimizes perplexity on training data

CSLM - Training



- Backprop training, cross-entropy error

$$E = \sum_{i=1}^{N} d_i \log p_i$$

+ weight decay

⇒ NN minimizes perplexity on training data

- continuous word codes are also learned (random initialization)

Recurrent Network for LM

Theoretical aspects

- Ideally, one should estimate:

  $$P(w_1^p) = P(w_1)\prod_{i=2}^{p} P(w_i|w_1^{i-1})$$

  i.e. each word is conditioned on all preceding words
- A recurrent neural network seems to be the perfect choice
- This was proposed by Mikolov et al in 2010, and many follow-up works

Recurrent Network for LM

Theoretical aspects

- Ideally, one should estimate:

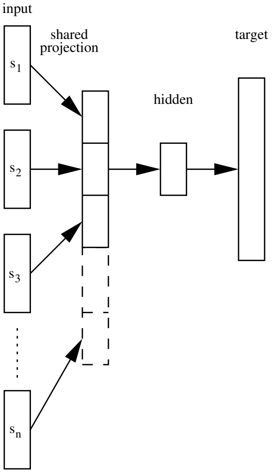
$$P(w_1^p) = P(w_1) \prod_{i=2}^{p} P(w_i | w_1^{i-1})$$

  i.e. each word is conditioned on all preceding words
- A recurrent neural network seems to be the perfect choice
- This was proposed by Mikolov et al in 2010, and many follow-up works

Practical issues

- Gradients tend to vanish for long sequences
  $\rightarrow$ Long Short-Term Memory (LSTM) networks
- It is less obvious to optimize recurrent NN

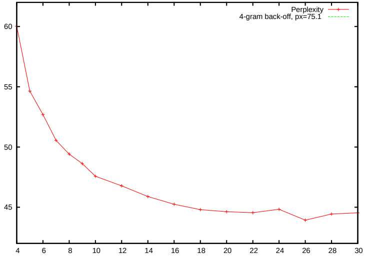
CSLM-MLP with larger Context Windows

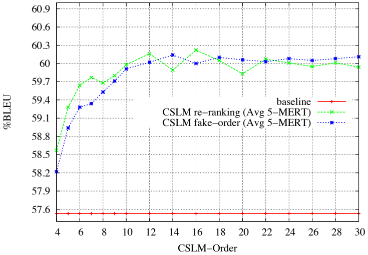


Advantages

- It is very easy to increase the size of the context window
- The number of parameter only increases slightly (the projections are shared)
- A special token `NULL_WORD` is used to handle shorter contexts
- The network always sees the full context, no vanishing of words which are far away
- But it may be more complicated to detect structure

CSLM-MLP with larger Context Windows





- The perplexity decreases significantly: 4g=60, 16g=45
- The NN clearly benefits from longer contexts
- Significant gain w/r to back-off LM: -40%
- These gains in perplexity carry over to improvements in the BLEU score

Continuous Space Translation Models

Motivation

- Can we apply similar ideas to the translation model ?
- Good probability estimation seems to be very important for the translation model
  - Appropriate bitexts will be always a sparse resources
  - We have many rare and unseen events

Estimating Phrase-Pair Probabilities

Definition (usually $p, q \in [1, 7]$):

$$P(\bar{\mathbf{t}}|\bar{\mathbf{s}}) = P(t_1 \dots t_p | s_1 \dots s_q)$$

This equation can be factorized as follows:

$$\begin{aligned}
& P(t_1, \dots, t_p | s_1, \dots, s_q) \\
& \quad = P(t_1 | t_2, \dots, t_p, s_1, \dots, s_q) \times P(t_2, \dots, t_p | s_1, \dots, s_q) \\
& \quad = P(t_1 | t_2, \dots, t_p, s_1, \dots, s_q) \times P(t_2 | t_3, \dots, t_p, s_1, \dots, s_q) \\
& \quad \quad \times P(t_3, \dots, t_p | s_1, \dots, s_q) \\
& \quad = \prod_{k=1}^{p} P(t_k | t_{k+1}, \dots, t_p, s_1, \dots, s_q) \\
& \quad \approx \prod_{k=1}^{p} P(t_k | s_1, \dots, s_q) = \prod_{k=1}^{p} P(t_k | \bar{\mathbf{s}})
\end{aligned}$$

Estimating Phrase-Pair Probabilities

$$P(t_1, \ldots, t_p | s_1, \ldots, s_q) = \prod_{k=1}^{p} P(t_k | t_{k+1}, \ldots, t_p, s_1, \ldots, s_q)$$

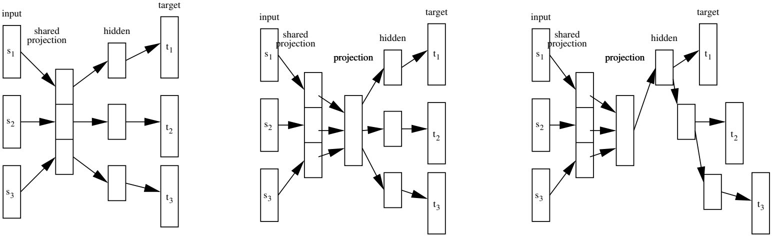
$$\approx \prod_{k=1}^{p} P(t_k | s_1, \ldots, s_q) = \prod_{k=1}^{p} P(t_k | \bar{\mathbf{s}})$$

- We drop the dependence between the target words

$\Rightarrow$ p independent "n-gram models" which try to predict the kth word in the target phrase given all the words of the source phrase $\bar{\mathbf{s}}$.

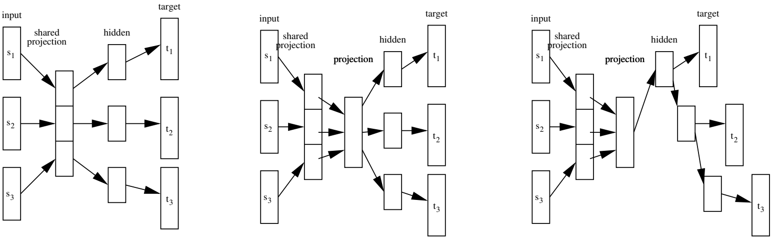
- Such a model is actually a generalization of the CSLM with multiple outputs (there are no constraints to use the same vocabulary at the input and the output of the NN)

CSTM Architectures



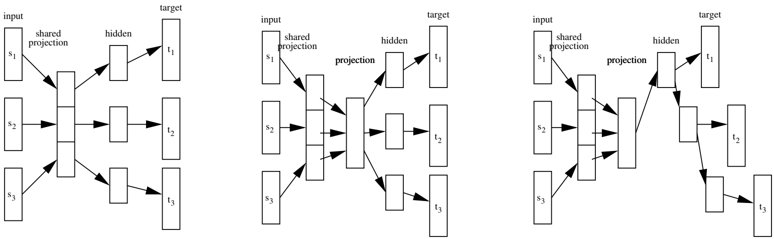
- Left: simple extension of the CSLM.

CSTM Architectures



- Left: simple extension of the CSLM.
- Middle: addition of a common hidden layer

CSTM Architectures



- Left: simple extension of the CSLM.
- Middle: addition of a common hidden layer
- Right: hierarchical dependence

Improved Neural Translation Models

Joint translation models

- Condition the next target word on the preceding target words and a source context
- Le et al, Continuous Space Translation Models with Neural Networks [NAACL'12]
- Auli et al, Joint Language and Translation Modeling with Recurrent Neural Networks [EMNLP'13]
- Devlin et al, Fast and Robust Neural Network Joint Models [ACL'14]
- All are integrated into an traditional phrase-based system
  - n-best rescoring
  - directly into the decoder (needs heavy optimisation)

⇒ Significant improvements

In all these approaches, there is only one word at the output not a sequence !

Fully Neural MT Systems

- Can we replace all the other parts of an phrase-based SMT system with neural networks ?
- There are indeed works to use neural networks for word alignment, MERT, etc

$\Rightarrow$ we still rely on the independent development of many components

- Let's get rid of everything and train one neural architecture end to end

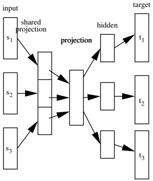
Neural Machine Translation

Main idea

- Continuous Space Translation Models [Schwenk, 2012]

  - N-gram approach to map some source words to some target words

  $\Rightarrow$ The joint layer in the middle encodes the phrases



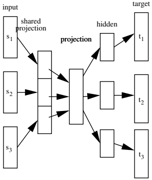
- How to generalize this ?

Neural Machine Translation

Main idea

- Continuous Space Translation Models [Schwenk, 2012]

  \- N-gram approach to map some source words to some target words
  $\Rightarrow$ The joint layer in the middle encodes the phrases



- How to generalize this ?
  - replace short phrases by entire sentences
  - condition next target word on preceding ones
  - use two recurrent NNs instead of an N-gram approach (at the input and the output)

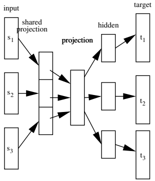
Neural Machine Translation

Main idea

- Continuous Space Translation Models [Schwenk, 2012]

  - N-gram approach to map some source words to some target words

  ⇒ The joint layer in the middle encodes the phrases



- How to generalize this ?
  - replace short phrases by entire sentences
  - condition next target word on preceding ones
  - use two recurrent NNs instead of an N-gram approach (at the input and the output)

⇒ Encoder/Decoder approach

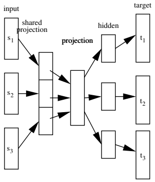
Neural Machine Translation

Main idea

- Continuous Space Translation Models [Schwenk, 2012]

  - N-gram approach to map some source words to some target words

  $\Rightarrow$ The joint layer in the middle encodes the phrases



- How to generalize this ?
  - replace short phrases by entire sentences
  - condition next target word on preceding ones
  - use two recurrent NNs instead of an N-gram approach (at the input and the output)

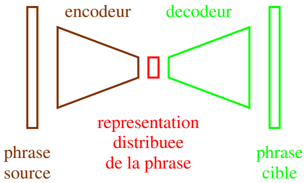
$\Rightarrow$ Encoder/Decoder approach

- also called Sequence-to-Sequence processing

Encoder/Decoder Approach

General idea

- An encoder processes the source sentence and creates an compact representation
- This representation is the input to the decoder which generates a sequence in the target sentence

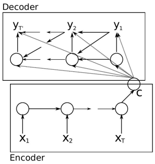


Encoder/Decoder Approach

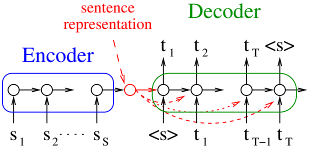
Instances of this idea (all published in 2014)

- *A Convolutional Neural Network for Modeling Sentences*
  Kalchbrenner et al, ACL, June 2014
  - encoder: convolutional n-gram model
  - decoder: hybrid of inverse convolutional model and RNN
  - rescoring of SMT n-best lists
- *Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation*
  Cho et al, EMNLP, Oct 2014
  - encoder/decoder: RNN with GRU
  - initially on phrases only and n-best rescoring
  - later applied to full sentences
- *Sequence to Sequence Learning with Neural Networks*
  Sutskever et al, NIPS, Dec 2014
  - encoder/decoder: huge stacked LSTM
  - full sentences

RNN Encoder–Decoder for SMT, Cho et al



(original figure from Cho et al.)



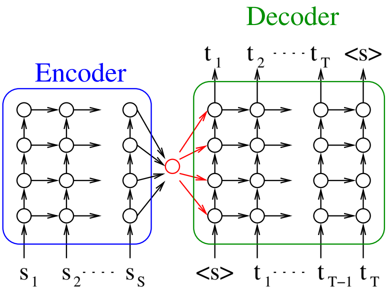
different representation

- Encoder/decoder: LSTM or GRU (gated recurrent NN)
- Encoder: no output layer, no loss function,
  → gradients are back-propagated from the decoder
- Initially used to calculate phrase translation probabilities (additional feature function in PBSMT system)

⇒ Improvement of 0.5 – 1 BLEU

- Generalized in follow-up work to a fully neural MT system (trained directly on bitexts)

Sequence-to-Sequence Processing



Some details:

- Huge LSTM: 4 layers with 1k neurons + vertical dropout
- Present source sentence in inverted order
- Beam size 12
- Rescoring an En/Fr SMT system: 33.30 → 35.61 / 36.5
- NMT alone: 30.59, ensemble of 5 systems: 34.81

Neural MT with Attention

Neural Machine Translation by Jointly Learning to Align and Translate, Bahdanau et al, ICLR 2015

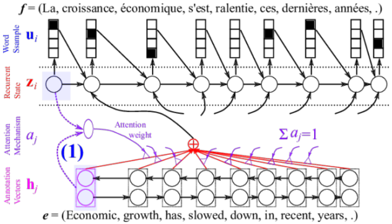
Main idea

- Do not attempt to memorize the whole sequence
- But condition each target word on a subset of source words

$\Rightarrow$ The neural network learns itself which source words are important to predict the next target word

- Notion of a soft alignment
- Automatic attention mechanism (also very successful in vision)

Neural MT with Attention



(figure from Bahdanau, Cho et al.)

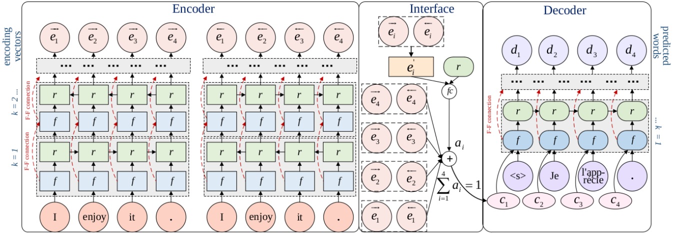
- Forward and backward LSTM

→ Sequence of concatenated vectors which summarizes past and future at a given time step

- Attention mechanism:
  - each target word is conditioned on a linear combination of these vectors
  - in practice, this (soft) attention is focused on few words
  - the weights a_j are also learned by the neural network

Very Deep Neural MT

Deep Recurrent Models with Fast-Forward Connections for Neural Machine Translation, Jie Zhou et al, TACL'16



- Alternation between forward and backward LSTMs (2 blocks)
- ResNet-style fast-forward connections
- Significant improvements over vanilla NMT: +6 BLEU
- Does also work quite well without attention (-1.4 BLEU)

Neural Machine Translation: Challenges

- Handling of the large output vocabulary and OOV
- What is the best basic unit: chars, subwords or words ?
- How to leverage unlabeled, eg. monolingual, data ?
- Deeper/other architectures for the encoder and decoder ?
- Mismatch between training criteria and inference
- Do we need the notion of coverage in the alignement model ?

$\Rightarrow$ Very active research field, continued improvements

- **But:** Hopefully, we won't start adding many "indendent hacks" to address various issues

Let's keep a globally optimized approach with a well defined criterion

Neural MT Joint Training

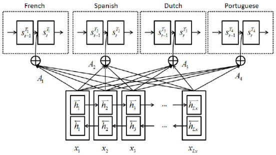
Motivation

- In the standard approach, systems for several language pairs are developed independently
- No sharing of resources, models and development work
- Translation from one source language into several target languages
  - can we leverage knowledge extraction to improve the encoder ?
  - particularly intersting with unbalanced resources
- Translation from several languages into one target language
  - improved decoder by sharing resources, training, etc ?

NMT Joint Training: One-to-Many

Multi-Task Learning for Multiple Language Translation, Dong et al,

- Translate the same source language into several target languages
- Attention model specific to each language pair



(figure from Dong et al.)

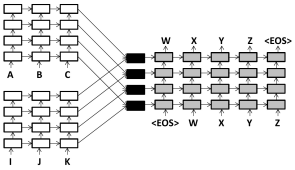
$\Rightarrow$ we can expect an improved encoder

- WMT task: English $\rightarrow$ ES/Fr/Nl and Pt
- Improvements of 0.5 – 1.4 BLEU
- Seems to help under-resourced language pairs

NMT Joint Training: Many-to-One

Multi-Source Neural Translation, Zopf and Knight, NAACL'16

- Translate text which is simultaneously available in two language to a third one ⇒ Needs a **trilingual** corpus
- Combine representation of two source languages:



(figure from Zopf and Knight)

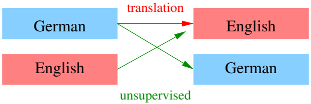
- WMT task: French + German → English: +4.8 BLEU

⇒ The input in different language is clearly complementary

NMT Joint Training: Many-to-Many

Multi-Task Sequence to Sequence Learning, Luong et al, ICLR'16

- Investigated three settings (no attention model)
  - One-to-Many: En $\rightarrow$ Ge / POS tags / En
  - Many-One-to: Ge / images / En $\rightarrow$ En
  - Many-to-Many: En / Ge $\rightarrow$ En / Ge

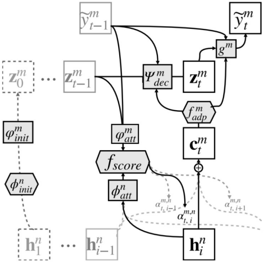


- Improvements are observed in all conditions
- Monolingual data:
  - auto-encoder didn't work
  - predict second half of sentence given the first one

NMT Joint Training: Many-to-Many

Multi-Way, Multilingual Neural Machine Translation with a Shared Attention Mechanism, Firat et al, NAACL'16

- Many-to-Many setting with attention mechanism
- The authors aim to use **one shared attention mechanism** for all language pairs
- Quite tricky and requires a complicated architecture



(figure from Orhan et al.)

- WMT task (4 languages): up to +1 BLEU point

Generalized NMT Joint Training

What do we want

- Jointly train on many languages and modalities
- Efficient way to use unlabeled data in the encoder and decoder
- Low-resource language pairs benefit from other parallel data
- Zero-shot translation

Generalized NMT Joint Training

What do we want

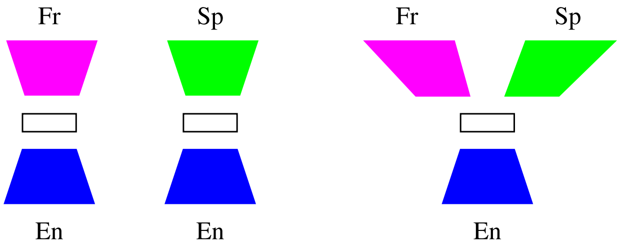
- Jointly train on many languages and modalities
- Efficient way to use unlabeled data in the encoder and decoder
- Low-resource language pairs benefit from other parallel data
- Zero-shot translation

Proposed architecture

- Joint language independent **sentence representation**

$\Rightarrow$ Continuous Interlingua

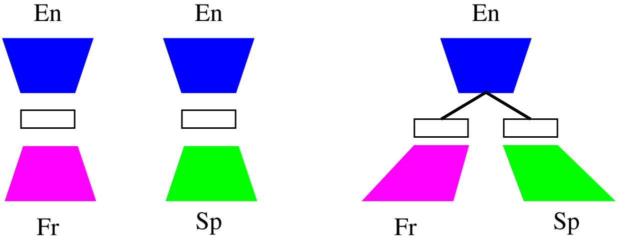
Generalized NMT Joint Training



Independently trained NMT systems

- Left figure: there is no reason that the same English sentence is represented the same way in both systems
- Use one joint encoder (right figure)
- Alternate between En/Fr and En/Sp data

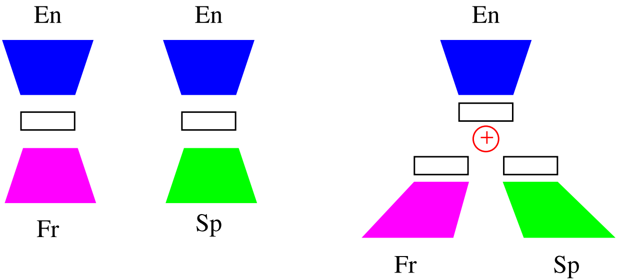
Generalized NMT Joint Training



Independently trained NMT systems

- Left figure: there is no reason that a sentence translated into the same English sentence is represented identical in the source representation
- Right figure: using one joint decoder **does not solve the problem !**

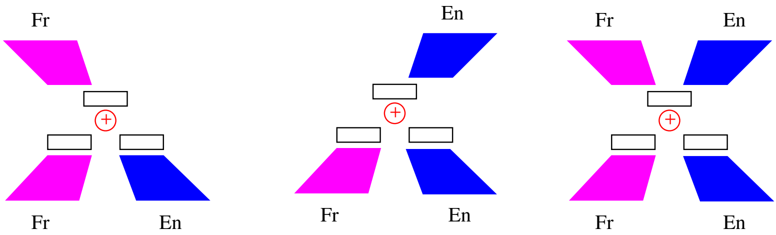
Generalized NMT Joint Training



Joint source representation

- We need to perform some operation to force both encoders to learn the same representation
- Do we need trilingual data to achieve this ?

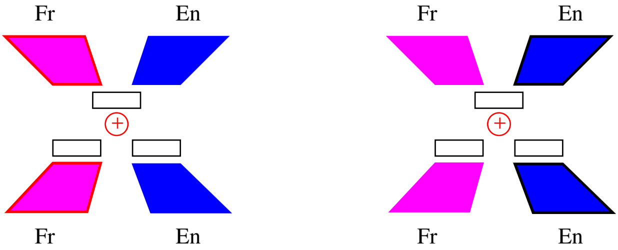
Generalized NMT Joint Training



Learning joint source representations with bitexts

- Put source and target at the input
- Translate into one of the languages, or both simultaneously

Generalized NMT Joint Training

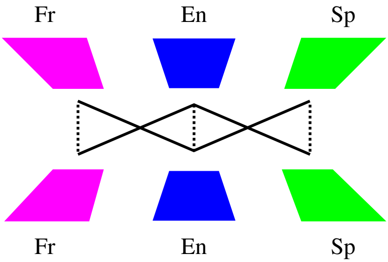


Training with labeled and unlabeled data

- The same architecture can be trained with monolingual data $\rightarrow$ sentence autoencoder (does not work well with attention !)

$\Rightarrow$ Alternate between many (partial) training paths in the architecture

Generalized NMT Joint Training



Zero-shot neural MT

- Train the architecture with En/Fr and En/Sp bitexts and monolingual data of all languages

$\Rightarrow$ Since we have one joint representation we should be able to perform Fr $\leftrightarrow$ Sp without using such bitexts !

- The joint representation is the **abstract pivot language**

Generalized NMT Joint Training

Pushing the idea to the limit

- Train jointly on many languages and modalities (images, speech, ...)
- Using bitexts and unlabeled data
- For some corpora, we also have *n*-wise parallel sentences (e.g. Europarl, TED, UN)

Generalized NMT Joint Training

Pushing the idea to the limit

- Train jointly on many languages and modalities (images, speech, ...)
- Using bitexts and unlabeled data
- For some corpora, we also have *n*-wise parallel sentences (e.g. Europarl, TED, UN)

$\Rightarrow$ The joint representation necessarily captures the **semantics** of the input

Continuous Sentence Representations

Background

- The notion of a continuous representation of **words** is well motivated and understood
- The situation is less clear for a **sequence of words**
  - should we use a fixed-size vector or an attention-based approach ?
  - what are the syntactic and semantic relations in the space of sentences ?

Continuous Sentence Representations

Background

- The notion of a continuous representation of **words** is well motivated and understood
- The situation is less clear for a **sequence of words**
  - should we use a fixed-size vector or an attention-based approach ?
  - what are the syntactic and semantic relations in the space of sentences ?

Usage of sentence representations

- Connects encoder and decoder in NMT
- Joint representation enables zero-shot translation
- Search and compare sentences

⋮

Continuous Sentence Representations

Fixed size

- Can we compress a whole sentence into one vector ?
- How the sentence length is encoded ?

+ Makes learning a joint representation easier
+ Enables comparison and search in that space

– Performance tends to decrease with sentence length

Continuous Sentence Representations

Fixed size

- Can we compress a whole sentence into one vector ?
- How the sentence length is encoded ?

+ Makes learning a joint representation easier
+ Enables comparison and search in that space
– Performance tends to decrease with sentence length

Variable size / with attention

+ Performs better on long sentences
+ Attention mechanisms are very successful in many other areas
+ Alignments are useful in practical applications
– The notion of a joint representation is tricky
– Complicates the comparison of sentences

FAIR: Facebook AI Research

Every day on Facebook

- 10 billion text messages are sent
- 2 billion pictures are uploaded
- several millions of new videos are published
- 1.5 billion searches ar econducted

FAIR vision: AI will mediate communication

FAIR: Facebook AI Research

Every day on Facebook

- 10 billion text messages are sent
- 2 billion pictures are uploaded
- several millions of new videos are published
- 1.5 billion searches ar econducted

FAIR vision: AI will mediate communication

- between people
    - feed ranking, suggestions, real-time translation, etc.

FAIR: Facebook AI Research

Every day on Facebook

- 10 billion text messages are sent
- 2 billion pictures are uploaded
- several millions of new videos are published
- 1.5 billion searches ar econducted

FAIR vision: AI will mediate communication

- between people
  - feed ranking, suggestions, real-time translation, etc.
- between people and the digital world
  - content search, Q&A, real-time dialog

FAIR: Facebook AI Research

FAIR: Overview

- $\approx$ 40 research scientists
- $\approx$ 20 research engineers
- NYC, MPK, Paris, Seattle
- still growing . . .

FAIR: Facebook AI Research

FAIR: Overview

- $\approx$ 40 research scientists
- $\approx$ 20 research engineers
- NYC, MPK, Paris, Seattle
- still growing ...

Some projects:

- Image captioning for the visual impaired
- Image analysis (hash tags, content, ...)
- Face recognition
- Video analysis
- Machine translation, Q&A

Conclusion

- Word embeddings are everywhere
- Very active and competitive research field
- Neural networks are *"invading"* traditional NLP conferences
- They achieve state-of-the-art or superior performance in many NLP applications
- Neural network LMs have basically replaced discrete approaches
- Neural MT is likely to replace phrase-based systems in the near future ...

Open Questions and Challenges

- How should we represent best entire sentences ?
  - one unique huge vector or attention-based ?

Open Questions and Challenges

- How should we represent best entire sentences ?
  - one unique huge vector or attention-based ?
- Should we keep words as the basic token ?
  - sub-word (BPE) or even characters ?
  - should we explicitly handle morphology ?

Open Questions and Challenges

- How should we represent best entire sentences ?
  - one unique huge vector or attention-based ?
- Should we keep words as the basic token ?
  - sub-word (BPE) or even characters ?
  - should we explicitly handle morphology ?
- Is there a live beyond LSTMs ?

Open Questions and Challenges

- How should we represent best entire sentences ?
  - one unique huge vector or attention-based ?
- Should we keep words as the basic token ?
  - sub-word (BPE) or even characters ?
  - should we explicitly handle morphology ?
- Is there a live beyond LSTMs ?
- How to integrate background knowledge ?

Open Questions and Challenges

- How should we represent best entire sentences ?
  - one unique huge vector or attention-based ?
- Should we keep words as the basic token ?
  - sub-word (BPE) or even characters ?
  - should we explicitly handle morphology ?
- Is there a live beyond LSTMs ?
- How to integrate background knowledge ?
- Solve tasks independently or use one big joint training ?