Trends and Challenges in Language Modeling for Speech Recognition and Machine Translation

Holger Schwenk

LIUM, University of Le Mans, France

Holger.Schwenk@lium.univ-lemans.fr

December 15, 2009

Trends and Challenges in Language Modeling

- Is there a live beyond back-off *n*-grams ?
- Will we modify Kneser-Ney smoothing again ?
- Will we be able to do research without relying on Google to provide large text collections ?
- How to obtain more research grants to buy more powerful computers ?

Trends and Challenges in Language Modeling

- Is there a live beyond back-off n-grams ?
- Will we modify Kneser-Ney smoothing again ?
- Will we be able to do research without relying on Google to provide large text collections ?
- How to obtain more research grants to buy more powerful computers ?

Trends and Challenges in Language Modeling

- Is there a live beyond back-off n-grams ?
- Will we modify Kneser-Ney smoothing again ?
- Will we be able to do research without relying on Google to provide large text collections ?
- How to obtain more research grants to buy more powerful computers ?

Trends and Challenges in Language Modeling

- Is there a live beyond back-off *n*-grams ?
- Will we modify Kneser-Ney smoothing again ?
- Will we be able to do research without relying on Google to provide large text collections ?
- How to obtain more research grants to buy more powerful computers ?

Trends and Challenges in Language Modeling

- Is there a live beyond back-off *n*-grams ?
- Will we modify Kneser-Ney smoothing again ?
- Will we be able to do research without relying on Google to provide large text collections ?
- How to obtain more research grants to buy more powerful computers ?

Applications of LM

- Automatic speech recognition (ASR)

$$\hat{w} = \arg\max_{w} Pr(w|x) = \arg\max_{w} Pr(w)Pr(x|w)$$

- Statistical machine translation (SMT), translate f to e

$$\hat{e} = \arg\max_{e} Pr(e|f) = \arg\max_{e} Pr(e)Pr(f|e)$$

Why should we invert the conditional probability ?

- We already have an LM since we have been working on ASR before
- The translation model is too bad and can't find good translations and smooth target sentence at once

Applications of LM

- Automatic speech recognition (ASR)

$$\hat{w} = \arg\max_{w} Pr(w|x) = \arg\max_{w} Pr(w)Pr(x|w)$$

- Statistical machine translation (SMT), translate f to e

$$\hat{e} = \arg\max_{e} Pr(e|f) = \arg\max_{e} Pr(e)Pr(f|e)$$

Why should we invert the conditional probability ?

- We already have an LM since we have been working on ASR before
- The translation model is too bad and can't find good translations and smooth target sentence at once

Applications of LM

- Automatic speech recognition (ASR)

$$\hat{w} = \arg\max_w Pr(w|x) = \arg\max_w Pr(w)Pr(x|w)$$

- Statistical machine translation (SMT), translate f to e

$$\hat{e} = \arg\max_e Pr(e|f) = \arg\max_e Pr(e)Pr(f|e)$$

Why should we invert the conditional probability ?

- We already have an LM since we have been working on ASR before
- The translation model is too bad and can't find good translations and smooth target sentence at once

Applications of LM

- Automatic speech recognition (ASR)

$$\hat{w} = \arg\max_{w} Pr(w|x) = \arg\max_{w} Pr(w)Pr(x|w)$$

- Statistical machine translation (SMT), translate f to e

$$\hat{e} = \arg\max_{e} Pr(e|f) = \arg\max_{e} Pr(e)Pr(f|e)$$

Why should we invert the conditional probability ?

- We already have an LM since we have been working on ASR before
- The translation model is too bad and can't find good translations and smooth target sentence at once

Applications of LM

- Automatic speech recognition (ASR)

$$\hat{w} = \arg\max_w Pr(w|x) = \arg\max_w Pr(w)Pr(x|w)$$

- Statistical machine translation (SMT), translate f to e

$$\hat{e} = \arg\max_e Pr(e|f) = \arg\max_e Pr(e)Pr(f|e)$$

Why should we invert the conditional probability ?

- We already have an LM since we have been working on ASR before
- The translation model is too bad and can't find good translations and smooth target sentence at once

Applications of LM

Speech Recognition

- The LM must choose among a large number of segmentations of the phoneme sequence into words, given the pronunciation lexicon
- The LM must also select among homonyms
- It deals with morphology (gender accordance, ...)
- The word order is given by the sequential processing of speech

Applications of LM

Machine translation

- Deal with morphology like for ASR
- The LM helps to choose between different translations
- Translation may require word reordering for certain language pairs

⇒ the LM has to sort out the good and the bad ones

Comparison

- It is an interesting question whether language modeling for MT is more or less difficult than for ASR
- One may consider that the semantic level is more important in MT

Applications of LM

Machine translation

- Deal with morphology like for ASR
- The LM helps to choose between different translations
- Translation may require word reordering for certain language pairs

$\Rightarrow$ the LM has to sort out the good and the bad ones

Comparison

- It is an interesting question whether language modeling for MT is more or less difficult than for ASR
- One may consider that the semantic level is more important in MT

Applications of LM

Example output of *good* SMT systems:

- *, it's a camera. I a do you have in Japan.* (BTEC Zh/En)
- *Oh, Japan produced by the camera than in Japan to buy cheaper ah. (Zh/En)*
- *Japanese strange, the camera here cheaper it in Japan.* (BTEC Ar/En)

Applications of LM

Example output of *good* SMT systems:

- *, it's a camera. I a do you have in Japan.* (BTEC Zh/En)
- *Oh, Japan produced by the camera than in Japan to buy cheaper ah. (Zh/En)*
- *Japanese strange, the camera here cheaper it in Japan.* (BTEC Ar/En)

Applications of LM

Example output of *good* SMT systems:

- *, it's a camera. I a do you have in Japan.* (BTEC Zh/En)
- *Oh, Japan produced by the camera than in Japan to buy cheaper ah. (Zh/En)*
- *Japanese strange, the camera here cheaper it in Japan.* (BTEC Ar/En)

Applications of LM to MT

Log-linear approach

$$\hat{e} = \arg\max_e Pr(e) Pr(f|e)$$
$$= \arg\max_e \prod_i Pr(e,f)^{\lambda_i}$$
$$= \arg\max_e \sum_i \lambda_i \log Pr(e,f)$$

λ_i are numerically optimized to maximize translation performance

- In practice, we use 5 scores for the translation model, a couple of scores for the reordering model a word penalty and one LM score

$\Rightarrow$ Apparently there is much more modeling effort on the TM than on the LM

Applications of LM to MT

Log-linear approach

$$
\begin{aligned}
\hat{e} &= \arg\max_e Pr(e) Pr(f|e) \\
&= \arg\max_e \prod_i Pr(e,f)^{\lambda_i} \\
&= \arg\max_e \sum_i \lambda_i \log Pr(e,f)
\end{aligned}
$$

λ_i are numerically optimized to maximize translation performance

- In practice, we use 5 scores for the translation model, a couple of scores for the reordering model a word penalty and one LM score

$\Rightarrow$ Apparently there is much more modeling effort on the TM than on the LM

Comparison of Research on LM

ASR		MT
3-gram back-off 4-gram back-off modif. KN	$\Rightarrow$	4-gram modif. KN
class LM linguistic motivated LMs Discriminative approaches	$\Rightarrow$	?
adaptation (MAP, IR + web)	$\Rightarrow$	starting slowly
2 papers	$\Leftarrow$	use of huge corpora distributed and compressed LMs

- MT has only taken over a small part of research from ASR
- Research on huge LMs seems to be limited to MT

Comparison of Research on LM

ASR		MT
3-gram back-off 4-gram back-off modif. KN	$\Rightarrow$	4-gram modif. KN
class LM linguistic motivated LMs Discriminative approaches	$\Rightarrow$	?
adaptation (MAP, IR + web)	$\Rightarrow$	starting slowly
2 papers	$\Leftarrow$	use of huge corpora distributed and compressed LMs

- MT has only taken over a small part of research from ASR
- Research on huge LMs seems to be limited to MT

Comparison of Research on LM

ASR		MT
3-gram back-off 4-gram back-off modif. KN	$\Rightarrow$	4-gram modif. KN
class LM linguistic motivated LMs Discriminative approaches	$\Rightarrow$	?
adaptation (MAP, IR + web)	$\Rightarrow$	starting slowly
2 papers	$\Leftarrow$	use of huge corpora distributed and compressed LMs

- MT has only taken over a small part of research from ASR
- Research on huge LMs seems to be limited to MT

Comparison of Research on AM and LM

Acoustic modeling (cf. talk of M. Gales)

- HMMs are still alive, but many new ideas
- Structure: decision tree state clustering
- Speaker adapation and adaptive training
- Discriminative methods, MMI, MCE, MPE, MPFE, ...
- Large margin approaches, ...

Language modeling

- A couple of papers at each conference
- Is the problem solved (with back-off n-grams) ?
- Did we give up ?

Comparison of Research on AM and LM

Acoustic modeling (cf. talk of M. Gales)

- HMMs are still alive, but many new ideas
- Structure: decision tree state clustering
- Speaker adapation and adaptive training
- Discriminative methods, MMI, MCE, MPE, MPFE, ...
- Large margin approaches, ...

Language modeling

- A couple of papers at each conference
- Is the problem solved (with back-off *n*-grams) ?
- Did we give up ?

Comparison of Research on AM and LM

Acoustic modeling (cf. talk of M. Gales)

- HMMs are still alive, but many new ideas
- Structure: decision tree state clustering
- Speaker adapation and adaptive training
- Discriminative methods, MMI, MCE, MPE, MPFE, ...
- Large margin approaches, ...

Language modeling

- A couple of papers at each conference
- Is the problem solved (with back-off n-grams) ?
- Did we give up ?

Comparison of Research on AM and LM

Acoustic modeling (cf. talk of M. Gales)

- HMMs are still alive, but many new ideas
- Structure: decision tree state clustering
- Speaker adapation and adaptive training
- Discriminative methods, MMI, MCE, MPE, MPFE, ...
- Large margin approaches, ...

Language modeling

- A couple of papers at each conference
- Is the problem solved (with back-off n-grams) ?
- Did we give up ?

No Data is better than more Data

Increasing amounts of data are available

- In-domain data (acoustic transcripts, bitexts): 100-200M
- Gigaword corpus: 1-3G words as a function of the language
- WEB data 100G -1T words

How to deal with so large amounts of data ?

- How to build the model ?
- How to store the model ?
- Hot to use the model ?

No Data is better than more Data

Increasing amounts of data are available

- In-domain data (acoustic transcripts, bitexts): 100-200M
- Gigaword corpus: 1-3G words as a function of the language
- WEB data 100G -1T words

How to deal with so large amounts of data ?

- How to build the model ?
- How to store the model ?
- Hot to use the model ?

No Data is better than more Data

Increasing amounts of data are available

- In-domain data (acoustic transcripts, bitexts): 100-200M
- Gigaword corpus: 1-3G words as a function of the language
- WEB data 100G -1T words *(this is 20 miles of books)*

How to deal with so large amounts of data ?

- How to build the model ?
- How to store the model ?
- Hot to use the model ?

No Data is better than more Data

Increasing amounts of data are available

- In-domain data (acoustic transcripts, bitexts): 100-200M
- Gigaword corpus: 1-3G words as a function of the language
- WEB data 100G -1T words

How to deal with so large amounts of data ?

- How to build the model ?
- How to store the model ?
- Hot to use the model ?

Very large Language Models

- IRSTLM
  [Federico et al, WMT'07]
- Distributed LM
  [Emami et al, ICASSP'07; Zhang et al, EMNLP'06]
- Stupid Back-off
  [Brants et al., EMNLP'07]
- Bloom Filter and randomized LMs
  [Talbot et al, EMNLP'07; ACL'07; ...]

IRSTLM

- M. Federico and M. Cettolo, *Efficient Handling of N-gram Language Models for Statistical Machine Translation*, WMT'07
- Clever data structures which focus on small memory usage
- Probability quantization
- LM is on one machine
- Experiments in SMT:
  - LM can be trained on more data, given a limited amount of main memory
  - This resulted in an increase of the translation performance

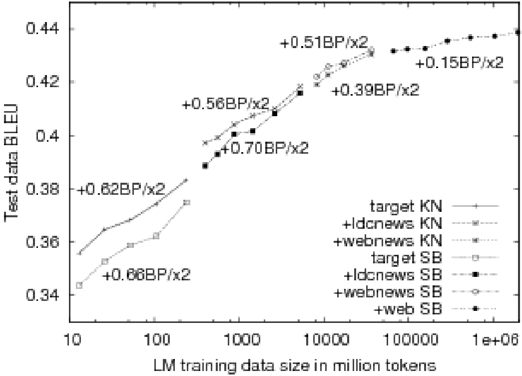
Distributed Language Models

- A. Emami, K. Papieni and J. Sorensen, *Large-Scale Distributed Language Modeling*, ICASSP'07
- Y. Zhang, A. Hildebrand and S. Vogel, *Distributed language modeling for n-best list reranking*, EMNLP'06
- LM is stored on multiple *LM workers*
- Data structure: suffix arrays
- Experiments in ASR:
  - Baseline 4-gram LM was trained on 192M words of in-domain data
  - Rescoring with distributed 5-gram trained on 4G words: +0.5% WER
- Experiments in MT:
  - Baseline 3-gram LM was trained on 2.8G words
  - Decoding with distributed 5-gram trained on 2.3G words: $\approx$ +3 points BLEU for Ar/En or Zh/En

Stupid Back-off

- T. Brants, A. Popat, P. Xu, F. Och and J. Dean, *Large Language Models in Machine Translation*, EMNLP'07
- Distributed storage of LM
- *Stupid Back-off smoothing* technique:
  directly use the relative frequencies and a fixed back-off weight
- Reorganziation of the MT search algorithm
- KN smoothed LMs were trained on up to 31G words
  (2 days on 400 machines, model size is 89GB)
- Stupid back-off was applied on up to 1.8T words
  (1 day on 1500 machines, model size is 1.8TB)

Stupid Back-off - Results for MT



- The authors report a steadily improvement of the translation quality as a function of the size of the LM training corpus

Google N-gram collection

Google made available a collection of 5-gram

- English (LDC 2006): 1.1G 5-grams from 1T words
- European languages (LDC 2009):
  100M words from 3 months in 2008

- Does anybody plan to use those for language modeling in ASR ?
- ASR people may be more concerned with speed than performance ?

Google N-gram collection

Google made available a collection of 5-gram

- English (LDC 2006): 1.1G 5-grams from 1T words
- European languages (LDC 2009): 100M words from 3 months in 2008

- Does anybody plan to use those for language modeling in ASR ?
- ASR people may be more concerned with speed than performance ?

Google N-gram collection

Google made available a collection of 5-gram

- English (LDC 2006): 1.1G 5-grams from 1T words
- European languages (LDC 2009): 100M words from 3 months in 2008

- Does anybody plan to use those for language modeling in ASR ?
- ASR people may be more concerned with speed than performance ?

Bloom Filters and Randomized LMs

- Lossy encoding based on Bloom filters: use of a data structure that sometimes makes an error, i.e. the model is unable to distinguish between distinct n-grams
- Two versions: store n-gram counts or probabilities in the Bloom filter
- Will always return the correct value for an n-gram that is in the model
- False positives: model can erroneously return a value for an n-gram that was never stored (in practice 0.0025%)
- Usually half the size of tree structure

What can we learn out of this ?

- Why huge LMs are mainly used in MT ?
- Is this a way to put semantic knowledge into the system ?
- *Every time I fire a linguist, the performance of our speech recognition system goes up* (Jelinek 1988)
- Should we now fire researchers and rather invest on data collection and more computers ?
- No, since there are many languages for which such large amounts of data are not (freely) available
- We can not always afford to work with huge distributed LMs: stand-alone PC systems, laptops, PDAs, smart phones
- It is less obvious to collect large amounts of data in other domains than "news", e.g. conversational or meeting speech, tourism related tasks, dictation devices (e.g. medical), military, . . .

What can we learn out of this ?

- Why huge LMs are mainly used in MT ?
- Is this a way to put semantic knowledge into the system ?
- *Every time I fire a linguist, the performance of our speech recognition system goes up* (Jelinek 1988)
- Should we now fire researchers and rather invest on data collection and more computers ?
- No, since there are many languages for which such large amounts of data are not (freely) available
- We can not always afford to work with huge distributed LMs: stand-alone PC systems, laptops, PDAs, smart phones
- It is less obvious to collect large amounts of data in other domains than "news", e.g. conversational or meeting speech, tourism related tasks, dictation devices (e.g. medical), military, ...

What can we learn out of this ?

- Why huge LMs are mainly used in MT ?
- Is this a way to put semantic knowledge into the system ?
- *Every time I fire a linguist, the performance of our speech recognition system goes up* (Jelinek 1988)
- Should we now fire researchers and rather invest on data collection and more computers ?
- No, since there are many languages for which such large amounts of data are not (freely) available
- We can not always afford to work with huge distributed LMs: stand-alone PC systems, laptops, PDAs, smart phones
- It is less obvious to collect large amounts of data in other domains than "news", e.g. conversational or meeting speech, tourism related tasks, dictation devices (e.g. medical), military, . . .

What can we learn out of this ?

- Why huge LMs are mainly used in MT ?
- Is this a way to put semantic knowledge into the system ?
- *Every time I fire a linguist, the performance of our speech recognition system goes up* (Jelinek 1988)
- Should we now fire researchers and rather invest on data collection and more computers ?
- No, since there are many languages for which such large amounts of data are not (freely) available
- We can not always afford to work with huge distributed LMs: stand-alone PC systems, laptops, PDAs, smart phones
- It is less obvious to collect large amounts of data in other domains than "news", e.g. conversational or meeting speech, tourism related tasks, dictation devices (e.g. medical), military, ...

What can we learn out of this ?

- Why huge LMs are mainly used in MT ?
- Is this a way to put semantic knowledge into the system ?
- *Every time I fire a linguist, the performance of our speech recognition system goes up* (Jelinek 1988)
- Should we now fire researchers and rather invest on data collection and more computers ?
- No, since there are many languages for which such large amounts of data are not (freely) available
- We can not always afford to work with huge distributed LMs: stand-alone PC systems, laptops, PDAs, smart phones
- It is less obvious to collect large amounts of data in other domains than "news", e.g. conversational or meeting speech, tourism related tasks, dictation devices (e.g. medical), military, ...

What can we learn out of this ?

- Why huge LMs are mainly used in MT ?
- Is this a way to put semantic knowledge into the system ?
- *Every time I fire a linguist, the performance of our speech recognition system goes up* (Jelinek 1988)
- Should we now fire researchers and rather invest on data collection and more computers ?
- No, since there are many languages for which such large amounts of data are not (freely) available
- We can not always afford to work with huge distributed LMs: stand-alone PC systems, laptops, PDAs, smart phones
- It is less obvious to collect large amounts of data in other domains than “news”, e.g. conversational or meeting speech, tourism related tasks, dictation devices (e.g. medical), military, ...

What can we learn out of this ?

- Why huge LMs are mainly used in MT ?
- Is this a way to put semantic knowledge into the system ?
- *Every time I fire a linguist, the performance of our speech recognition system goes up* (Jelinek 1988)
- Should we now fire researchers and rather invest on data collection and more computers ?
- No, since there are many languages for which such large amounts of data are not (freely) available
- We can not always afford to work with huge distributed LMs: stand-alone PC systems, laptops, PDAs, smart phones
- It is less obvious to collect large amounts of data in other domains than "news", e.g. conversational or meeting speech, tourism related tasks, dictation devices (e.g. medical), military, ...

What can we learn out of this ?

- Why huge LMs are mainly used in MT ?
- Is this a way to put semantic knowledge into the system ?
- *Every time I fire a linguist, the performance of our speech recognition system goes up* (Jelinek 1988)
- Should we now fire researchers and rather invest on data collection and more computers ?
- No, since there are many languages for which such large amounts of data are not (freely) available
- We can not always afford to work with huge distributed LMs: stand-alone PC systems, laptops, PDAs, smart phones
- It is less obvious to collect large amounts of data in other domains than "news", e.g. conversational or meeting speech, tourism related tasks, dictation devices (e.g. medical), military, . . .

Building LMs on small amounts of Data

Possible research directions

- Better smoothing ?
- Integration of syntactical or semantic knowledge ?
- Discriminative approaches ?
- Adaptation from a generic (news) model to a task specific one ?
- ...

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

Continuous Space LM

Theoretical drawbacks of back-off LM:

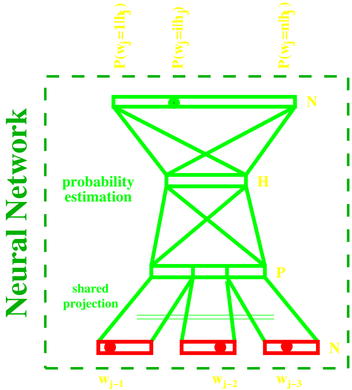
- Words are represented in a high-dimensional discrete space
- Probability distributions are not smooth functions
- Any change of the word indices can result in an arbitrary change of LM probability

$\Rightarrow$ 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

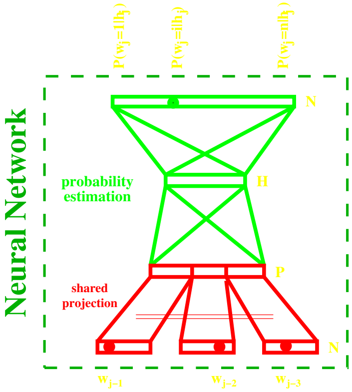
CSLM - Probability Calculation



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

- 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

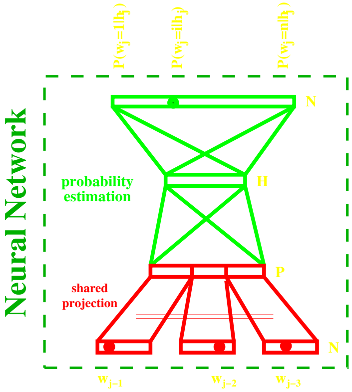
CSLM - Probability Calculation



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

- 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

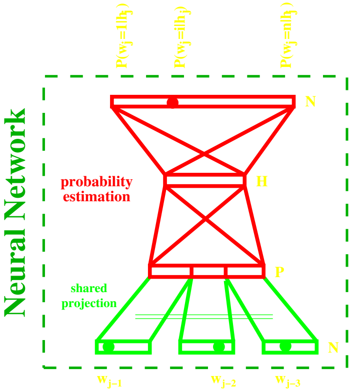
CSLM - Probability Calculation



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

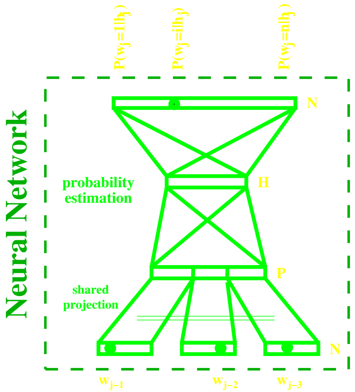
- Outputs = LM posterior probabilities of all words: $P(w_j = i | h_j) \quad \forall i \in [1, N]$
- Context h_j = sequence of $n-1$ points in this space
- Word = point in the P dimensional space
- Projection onto continuous space
- Inputs = indices of the $n-1$ previous words

CSLM - Probability Calculation



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

- 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

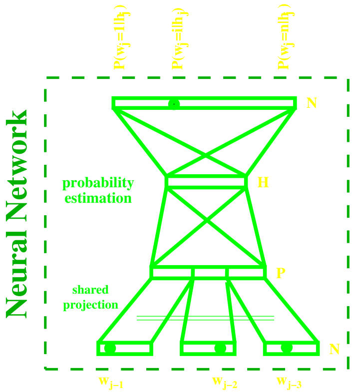
$$E = \sum_{i=1}^{N} d_i \log p_i$$

+ weight decay

⇒ NN minimizes perplexity on training data

- continuous word codes are also learned (random initialization)

CSLM - Training



- Backprop training, cross-entropy error

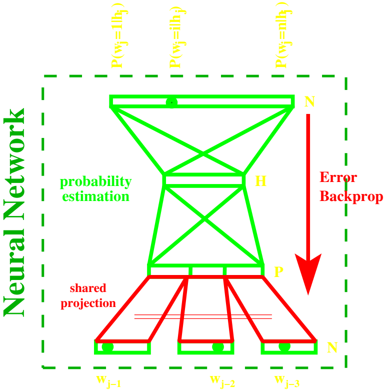
$$E = \sum_{i=1}^{N} d_i \log p_i$$

+ weight decay

$\Rightarrow$ NN minimizes perplexity on training data

- continuous word codes are also learned (random initialization)

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

- continuous word codes are also learned (random initialization)

Continuous Space LM

Some details (Computer Speech and Language, pp 492–518, 2007)

- Projection and estimation is done with a multi-layer neural network
- Still an n-gram approach, but an LM probability can be calculated for any n-gram without backing off
- Can be trained on the same data than the back-off LM using a resampling algorithm
- Efficient implementation is very important
- Used in lattice or n-best list rescoring

CSLM : Some Results in ASR

	Back-off LM WER	CSLM WER
En CTS	16.0%	15.5%
Ar CTS	30.8%	29.7%
En BN	9.6%	9.2%
Fr BN	10.7%	10.2%
En TC-Star	10.14%	9.17%
Sp TC-Star	7.55%	7.00%
En meetings	26.0%	24.4%
Ar Gale	13.7%	13.0%
Zh Gale	10.5%	10.1%

$\Rightarrow$ Improvements of 0.4 to 1.6% absolute

CSLM : Some Results in SMT

- BLEU scores on test data (the higher the better):

Task	Languages	#words	Back-off LM	CSLM
BTEC	It/En	200k	35.55	37.41
	Ar/En	200k	23.72	24.86
	Zh/En	400k	19.74	21.01
	Ja/En	400k	15.11	15.73
NIST	Ar/En	3.3G	47.02	47.90

- Significant improvements despite large amounts of LM training data (3.3G words)
- This gain corresponds to roughly 4x more training data
- Dealing with word order seems to be more challenging (Chinese and Japanese)

CSLM : Some Results in SMT

- BLEU scores on test data (the higher the better):

Task	Languages	#words	Back-off LM	CSLM
BTEC	It/En	200k	35.55	37.41
	Ar/En	200k	23.72	24.86
	Zh/En	400k	19.74	21.01
	Ja/En	400k	15.11	15.73
NIST	Ar/En	3.3G	47.02	47.90

- Significant improvements despite large amounts of LM training data (3.3G words)
- This gain corresponds to roughly 4x more training data
- Dealing with word order seems to be more challenging (Chinese and Japanese)

CSLM : Some Results in SMT

- BLEU scores on test data (the higher the better):

Task	Languages	#words	Back-off LM	CSLM
BTEC	It/En	200k	35.55	37.41
	Ar/En	200k	23.72	24.86
	Zh/En	400k	19.74	21.01
	Ja/En	400k	15.11	15.73
NIST	Ar/En	3.3G	47.02	47.90

- Significant improvements despite large amounts of LM training data (3.3G words)
- This gain corresponds to roughly 4x more training data
- Dealing with word order seems to be more challenging (Chinese and Japanese)

CSLM : Some Results in SMT

- BLEU scores on test data (the higher the better):

Task	Languages	#words	Back-off LM	CSLM
BTEC	It/En	200k	35.55	37.41
	Ar/En	200k	23.72	24.86
	Zh/En	400k	19.74	21.01
	Ja/En	400k	15.11	15.73
NIST	Ar/En	3.3G	47.02	47.90

- Significant improvements despite large amounts of LM training data (3.3G words)
- This gain corresponds to roughly 4x more training data
- Dealing with word order seems to be more challenging (Chinese and Japanese)

Continuous Space LM - Use

- Despite the good results the CSLM is not widely used
  - IBM has done several experiments in this direction
    New paper at this conference
  - Cambridge has recently reimplemented this approach

Continuous Space LM

Open source version

- Written in C++
- Interfaced with SRILM (uses same vocabularies, back-off LMs for short-lists and interpolation, ...)
- Fast NN training
(bunch mode, multi-threading, resampling, ...)
- *n*-best (and lattice) list rescoring
- Parameter tuning with Condor tool
- Download mid-January from
`http://liumtools.univ-lemans.fr`

$\Rightarrow$ Hopefully larger community will use and extend this approach

Outlook

- Don't try to memorize the whole world
- Keep low or medium size resourced tasks
- Try to put more structure into the models
- Discriminative and adaptive approaches, in particular for SMT
- Use and improve CSLM

Outlook

- Don't try to memorize the whole world
- Keep low or medium size resourced tasks
- Try to put more structure into the models
- Discriminative and adaptive approaches, in particular for SMT
- Use and improve CSLM

Outlook

- Don't try to memorize the whole world
- Keep low or medium size resourced tasks
- Try to put more structure into the models
- Discriminative and adaptive approaches, in particular for SMT
- Use and improve CSLM

Outlook

- Don't try to memorize the whole world
- Keep low or medium size resourced tasks
- Try to put more structure into the models
- Discriminative and adaptive approaches, in particular for SMT
- Use and improve CSLM

Outlook

- Don't try to memorize the whole world
- Keep low or medium size resourced tasks
- Try to put more structure into the models
- Discriminative and adaptive approaches, in particular for SMT
- Use and improve CSLM

Outlook

- Don't try to memorize the whole world
- Keep low or medium size resourced tasks
- Try to put more structure into the models
- Discriminative and adaptive approaches, in particular for SMT
- Use and improve CSLM