The impact of language models and loss functions on repair disfluency detection

Simon Zwarts and Mark Johnson
Department of Computing
Macquarie University
Mark.Johnson@mq.edu.au

June 3, 2011

Outline

Detecting and correcting speech errors in fluent speech

Previous work on disfluency detection

Language models and reranker features

Loss functions

Experimental results

Conclusion

A typology of speech disfluencies

- **Filled pauses:**

  I think it's uh refreshing to see the uh support ...

- **Parentheticals**

  But you know I was reading the other day ...

- **Repairs:**

  I want a flight to Boston uh I mean to Denver on Friday

- **Restarts:**

  Why didn't he why didn't she stay at home?

Bear, Dowding and Schriberg (1992), Heeman and Allen (1997, 1999), Nakatani and Hirschberg (1994), Stolcke and Schriberg (1996)

A typology of speech disfluencies

- **Filled pauses:**

  I think it's uh refreshing to see the uh support ...

- **Parentheticals**

  But you know I was reading the other day ...

- **Repairs:**

  I want a flight to Boston uh I mean to Denver on Friday

- **Restarts:**

  Why didn't he why didn't she stay at home?

Bear, Dowding and Schriberg (1992), Heeman and Allen (1997, 1999), Nakatani and Hirschberg (1994), Stolcke and Schriberg (1996)

A typology of speech disfluencies

- **Filled pauses:**

  I think it's uh refreshing to see the uh support ...

- **Parentheticals**

  But you know I was reading the other day ...

- **Repairs:**

  I want a flight to Boston uh I mean to Denver on Friday

- **Restarts:**

  Why didn't he why didn't she stay at home?

Bear, Dowding and Schriberg (1992), Heeman and Allen (1997, 1999), Nakatani and Hirschberg (1994), Stolcke and Schriberg (1996)

A typology of speech disfluencies

- **Filled pauses:**

  I think it's uh refreshing to see the uh support ...

- **Parentheticals**

  But you know I was reading the other day ...

- **Repairs:**

  I want a flight to Boston uh I mean to Denver on Friday

- **Restarts:**

  Why didn't he why didn't she stay at home?

Bear, Dowding and Schriberg (1992), Heeman and Allen (1997, 1999), Nakatani and Hirschberg (1994), Stolcke and Schriberg (1996)

Why treat restarts and repairs specially?

- Filled pauses are easy to recognise and remove *from speech transcripts*
- Modern NLP tools (e.g., parsers) handle parentheticals properly
- But *restarts and repairs* are often *misanalysed* by NLP tools

⇒ Detect and remove disfluencies before further processing

I want a flight to Boston uh I mean to Denver on Friday
Why didn't he why didn't she stay at home?

MACQUARIE UNIVERSITY

Why treat restarts and repairs specially?

- Filled pauses are easy to recognise and remove *from speech transcripts*
- Modern NLP tools (e.g., parsers) handle parentheticals properly
- But *restarts and repairs* are often *misanalysed* by NLP tools

$\Rightarrow$ Detect and remove disfluencies before further processing

I want a flight to Boston uh I mean to Denver on Friday

Why didn't he why didn't she stay at home?

MACQUARIE UNIVERSITY

Why treat restarts and repairs specially?

- Filled pauses are easy to recognise and remove *from speech transcripts*
- Modern NLP tools (e.g., parsers) handle parentheticals properly
- But *restarts and repairs* are often *misanalysed* by NLP tools

⇒ Detect and remove disfluencies before further processing

I want a flight to Boston uh I mean to Denver on Friday
Why didn't he why didn't she stay at home?

MACQUARIE UNIVERSITY

Why treat restarts and repairs specially?

- Filled pauses are easy to recognise and remove *from speech transcripts*
- Modern NLP tools (e.g., parsers) handle parentheticals properly
- But *restarts and repairs* are often *misanalysed* by NLP tools

⇒ Detect and remove disfluencies before further processing

I want a flight to Boston uh I mean to Denver on Friday

Why didn't he why didn't she stay at home?

MACQUARIE UNIVERSITY

The structure of restarts and repairs

$$\ldots\text{and}\ \underbrace{\text{you get,}}_{\text{Reparandum}}\ \underbrace{\text{uh,}}_{\text{Interregnum}}\ \underbrace{\text{you can get}}_{\text{Repair}}\ \text{a system}\ \ldots$$

- The Reparandum is *often not a syntactic phrase*
- The Interregnum is usually lexically and prosodically marked, but can be empty
- The Reparandum is often a *"rough copy"* of the Repair
  - Repairs are typically short
  - Repairs are not always copies
  - It's possible e.g. for there to be anaphoric dependencies into the reparandum

Shriberg 1994 "Preliminaries to a Theory of Speech Disfluencies"

MACQUARIE UNIVERSITY

The structure of restarts and repairs

$$\ldots\text{and}\ \underbrace{\text{you get,}}_{\text{Reparandum}}\ \underbrace{\text{uh,}}_{\text{Interregnum}}\ \underbrace{\text{you can get}}_{\text{Repair}}\ \text{a system}\ \ldots$$

- The Reparandum is *often not a syntactic phrase*
- The Interregnum is usually lexically and prosodically marked, but can be empty
- The Reparandum is often a *"rough copy"* of the Repair
  - Repairs are typically short
  - Repairs are not always copies
  - It's possible e.g. for there to be anaphoric dependencies into the reparandum

Shriberg 1994 "Preliminaries to a Theory of Speech Disfluencies"

MACQUARIE UNIVERSITY

The structure of restarts and repairs

$$\ldots\text{and } \underbrace{\text{you get,}}_{\text{Reparandum}} \; \underbrace{\text{uh,}}_{\text{Interregnum}} \; \underbrace{\text{you can get}}_{\text{Repair}} \text{ a system } \ldots$$

- The Reparandum is *often not a syntactic phrase*
- The Interregnum is usually lexically and prosodically marked, but can be empty
- The Reparandum is often a *"rough copy"* of the Repair
  - Repairs are typically short
  - Repairs are not always copies
  - It's possible e.g. for there to be anaphoric dependencies into the reparandum

Shriberg 1994 "Preliminaries to a Theory of Speech Disfluencies"

MACQUARIE UNIVERSITY

The structure of restarts and repairs

$$\ldots\text{and } \underbrace{\text{you get,}}_{\text{Reparandum}} \quad \underbrace{\text{uh,}}_{\text{Interregnum}} \quad \underbrace{\text{you can get}}_{\text{Repair}} \text{ a system } \ldots$$

- The Reparandum is *often not a syntactic phrase*
- The Interregnum is usually lexically and prosodically marked, but can be empty
- The Reparandum is often a *"rough copy"* of the Repair
  - Repairs are typically short
  - Repairs are not always copies
  - It's possible e.g. for there to be anaphoric dependencies into the reparandum

Shriberg 1994 "Preliminaries to a Theory of Speech Disfluencies"

MACQUARIE UNIVERSITY

The structure of restarts and repairs

$$\ldots\text{and } \underbrace{\text{you get,}}_{\text{Reparandum}} \quad \underbrace{\text{uh,}}_{\text{Interregnum}} \quad \underbrace{\text{you can get}}_{\text{Repair}} \text{ a system } \ldots$$

- The Reparandum is *often not a syntactic phrase*
- The Interregnum is usually lexically and prosodically marked, but can be empty
- The Reparandum is often a *"rough copy"* of the Repair
  - Repairs are typically short
  - Repairs are not always copies
  - It's possible e.g. for there to be anaphoric dependencies into the reparandum

Shriberg 1994 "Preliminaries to a Theory of Speech Disfluencies"

MACQUARIE UNIVERSITY

The structure of restarts and repairs

$$\ldots\text{and } \underbrace{\text{you get,}}_{\text{Reparandum}} \quad \underbrace{\text{uh,}}_{\text{Interregnum}} \quad \underbrace{\text{you can get}}_{\text{Repair}} \text{ a system} \ldots$$

- The Reparandum is *often not a syntactic phrase*
- The Interregnum is usually lexically and prosodically marked, but can be empty
- The Reparandum is often a *"rough copy"* of the Repair
  - Repairs are typically short
  - Repairs are not always copies
  - It's possible e.g. for there to be anaphoric dependencies into the reparandum

Shriberg 1994 "Preliminaries to a Theory of Speech Disfluencies"

Outline

Detecting and correcting speech errors in fluent speech

Previous work on disfluency detection

Language models and reranker features

Loss functions

Experimental results

Conclusion

Machine-learning approaches to disfluency detection

$$\ldots\ \overset{\_}{\text{and}}\ \underbrace{\overset{\text{E}}{\text{you}}\ \ \overset{\text{E}}{\text{get}}}_{\text{Reparandum}}\ \ \underbrace{\overset{\text{E}}{\text{uh}}}_{\text{Interregnum}}\ \ \underbrace{\overset{\_}{\text{you}}\ \ \overset{\_}{\text{can}}\ \ \overset{\_}{\text{get}}}_{\text{Repair}}\ \ \overset{\_}{\text{a}}\ \ \overset{\_}{\text{system}}\ \ldots$$

- Train a classifier to predict whether each word is EDITED or NOTEDITED
  - this approach classifies each word independently, but the classification should really be made over groups of words
- A *very large number of features* can be usefully deployed in such a system

Charniak and Johnson (2001), Zhang, Weng and Feng (2006)

Machine-learning approaches to disfluency detection

$$\ldots\ \overset{\_}{\text{and}}\ \underbrace{\overset{\text{E}}{\text{you}}\ \overset{\text{E}}{\text{get}}}_{\text{Reparandum}}\ \underbrace{\overset{\text{E}}{\text{uh}}}_{\text{Interregnum}}\ \underbrace{\overset{\_}{\text{you}}\ \overset{\_}{\text{can}}\ \overset{\_}{\text{get}}}_{\text{Repair}}\ \overset{\_}{\text{a}}\ \overset{\_}{\text{system}}\ \ldots$$

- Train a classifier to predict whether each word is EDITED or NOTEDITED
  - this approach classifies each word independently, but the classification should really be made over groups of words
- A *very large number of features* can be usefully deployed in such a system

Charniak and Johnson (2001), Zhang, Weng and Feng (2006)

MACQUARIE UNIVERSITY

Machine-learning approaches to disfluency detection

$$\ldots\ \overset{\_}{\text{and}}\ \underbrace{\overset{\text{E}}{\text{you}}\ \ \overset{\text{E}}{\text{get}}}_{\text{Reparandum}}\ \ \underbrace{\overset{\text{E}}{\text{uh}}}_{\text{Interregnum}}\ \ \underbrace{\overset{\_}{\text{you}}\ \ \overset{\_}{\text{can}}\ \ \overset{\_}{\text{get}}}_{\text{Repair}}\ \ \overset{\_}{\text{a}}\ \ \overset{\_}{\text{system}}\ \ldots$$

- Train a classifier to predict whether each word is EDITED or NOTEDITED
  - this approach classifies each word independently, but the classification should really be made over groups of words
- A *very large number of features* can be usefully deployed in such a system

Charniak and Johnson (2001), Zhang, Weng and Feng (2006)

MACQUARIE UNIVERSITY

The "true" model of repairs (?)

$\ldots$ and $\underbrace{\text{you get,}}_{\text{Reparandum}}$ $\underbrace{\text{uh,}}_{\text{Interregnum}}$ $\underbrace{\text{you can get}}_{\text{Repair}}$ a system $\ldots$

- Speaker generates intended "conceptual representation"
- Speaker incrementally generates syntax and phonology,
  - recognizes that what is said doesn't mean what was intended,
  - "backs up", i.e., partially deconstructs syntax and phonology, and
  - starts incrementally generating syntax and phonology again
- (but without a good model of "conceptual representation", this may be hard to formalize . . .)

The "true" model of repairs (?)

$$\text{...and } \underbrace{\text{you get,}}_{\text{Reparandum}} \; \underbrace{\text{uh,}}_{\text{Interregnum}} \; \underbrace{\text{you can get}}_{\text{Repair}} \text{ a system ...}$$

- Speaker generates intended "conceptual representation"
- Speaker incrementally generates syntax and phonology,
  - recognizes that what is said doesn't mean what was intended,
  - "backs up", i.e., partially deconstructs syntax and phonology, and
  - starts incrementally generating syntax and phonology again
- (but without a good model of "conceptual representation", this may be hard to formalize ...)

The "true" model of repairs (?)

$$\ldots\text{and } \underbrace{\text{you get,}}_{\text{Reparandum}} \; \underbrace{\text{uh,}}_{\text{Interregnum}} \; \underbrace{\text{you can get}}_{\text{Repair}} \text{ a system } \ldots$$

- Speaker generates intended "conceptual representation"
- Speaker incrementally generates syntax and phonology,
  - recognizes that what is said doesn't mean what was intended,
  - "backs up", i.e., partially deconstructs syntax and phonology, and
  - starts incrementally generating syntax and phonology again
- (but without a good model of "conceptual representation", this may be hard to formalize ...)

MACQUARIE UNIVERSITY

The "true" model of repairs (?)

$$\ldots\text{and } \underbrace{\text{you get,}}_{\text{Reparandum}}\ \underbrace{\text{uh,}}_{\text{Interregnum}}\ \underbrace{\text{you can get}}_{\text{Repair}} \text{ a system } \ldots$$

- Speaker generates intended "conceptual representation"
- Speaker incrementally generates syntax and phonology,
  - recognizes that what is said doesn't mean what was intended,
  - "backs up", i.e., partially deconstructs syntax and phonology, and
  - starts incrementally generating syntax and phonology again
- (but without a good model of "conceptual representation", this may be hard to formalize . . .)

The "true" model of repairs (?)

$\ldots$ and $\underbrace{\text{you get,}}_{\text{Reparandum}}$ $\underbrace{\text{uh,}}_{\text{Interregnum}}$ $\underbrace{\text{you can get}}_{\text{Repair}}$ a system $\ldots$

- Speaker generates intended "conceptual representation"
- Speaker incrementally generates syntax and phonology,
  - recognizes that what is said doesn't mean what was intended,
  - "backs up", i.e., partially deconstructs syntax and phonology, and
  - starts incrementally generating syntax and phonology again
- (but without a good model of "conceptual representation", this may be hard to formalize ...)

The "true" model of repairs (?)

$$\ldots \text{and } \underbrace{\text{you get,}}_{\text{Reparandum}} \quad \underbrace{\text{uh,}}_{\text{Interregnum}} \quad \underbrace{\text{you can get}}_{\text{Repair}} \text{ a system } \ldots$$

- Speaker generates intended "conceptual representation"
- Speaker incrementally generates syntax and phonology,
  - recognizes that what is said doesn't mean what was intended,
  - "backs up", i.e., partially deconstructs syntax and phonology, and
  - starts incrementally generating syntax and phonology again
- (but without a good model of "conceptual representation", this may be hard to formalize . . .)

MACQUARIE UNIVERSITY

Approximating the "true model"

I want a flight to Boston uh I mean to Denver on Friday

- Use Repair string as approximation to intended meaning
- Reparandum string is "rough copy" of Repair string
  - involves *crossing* (rather than *nested*) dependencies
- String with reparandum and interregnum excised is usually well-formed
  - after correcting the error, what's left should have high probability

  ⇒ *use model of normal language to interpret ill-formed input*

MACQUARIE UNIVERSITY

Approximating the "true model"

I want a flight to Boston uh I mean to Denver on Friday

- Use Repair string as approximation to intended meaning
- Reparandum string is "rough copy" of Repair string
  - involves *crossing* (rather than *nested*) dependencies
- String with reparandum and interregnum excised is usually well-formed
  - after correcting the error, what's left should have high probability
  - $\Rightarrow$ *use model of normal language to interpret ill-formed input*

Approximating the "true model"

I want a flight to Boston uh I mean to Denver on Friday

- Use Repair string as approximation to intended meaning
- Reparandum string is "rough copy" of Repair string
  - involves *crossing* (rather than *nested*) dependencies
- String with reparandum and interregnum excised is usually well-formed
  - after correcting the error, what's left should have high probability
  - ⇒ *use model of normal language to interpret ill-formed input*

MACQUARIE UNIVERSITY

Approximating the "true model"

I want a flight to Boston uh I mean to Denver on Friday

- Use Repair string as approximation to intended meaning
- Reparandum string is "rough copy" of Repair string
  - involves *crossing* (rather than *nested*) dependencies
- String with reparandum and interregnum excised is usually well-formed
  - after correcting the error, what's left should have high probability

$\Rightarrow$ *use model of normal language to interpret ill-formed input*

Approximating the "true model"

I want a flight to Boston uh I mean to Denver on Friday

- Use Repair string as approximation to intended meaning
- Reparandum string is "rough copy" of Repair string
  - involves *crossing* (rather than *nested*) dependencies
- String with reparandum and interregnum excised is usually well-formed
  - after correcting the error, what's left should have high probability

$\Rightarrow$ *use model of normal language to interpret ill-formed input*

Approximating the "true model"

I want a flight to Boston uh I mean to Denver on Friday

- Use Repair string as approximation to intended meaning
- Reparandum string is "rough copy" of Repair string
  - involves *crossing* (rather than *nested*) dependencies
- String with reparandum and interregnum excised is usually well-formed
  - after correcting the error, what's left should have high probability

$\Rightarrow$ *use model of normal language to interpret ill-formed input*

MACQUARIE UNIVERSITY

The Noisy Channel Model

Source model $\Pr(X)$
(statistical parser)

Source signal x
... and you can get a system ...

Noisy channel model $\Pr(U|X)$

Noisy signal u
... and you get, you can get a system ...

- Noisy channel models combines two different submodels
- Channel model needs to generate *crossing dependencies*
  $\Rightarrow$ TAG transducer

Johnson and Charniak (2004)

The Noisy Channel Model

Source model $\Pr(X)$
(statistical parser)

↓

Source signal x
. . . and you can get a system . . .

↓

Noisy channel model $\Pr(U|X)$

↓

Noisy signal u
. . . and you get, *you can get a system . . .*

- Noisy channel models combines two different submodels
- Channel model needs to generate *crossing dependencies*
  $\Rightarrow$ TAG transducer

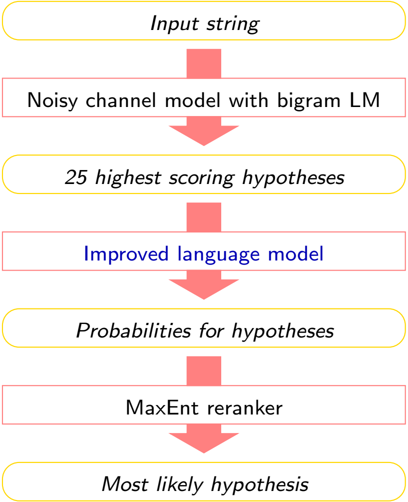
Johnson and Charniak (2004)

Reranking the Noisy Channel model

- Log probs from source model and channel model are *reranker features*
- MaxEnt reranker can use *additional features* as well

⇒ Best of both noisy channel and machine-learning approaches

- Johnson et al used a parser-based language model



Johnson, Charniak and Lease (2004)

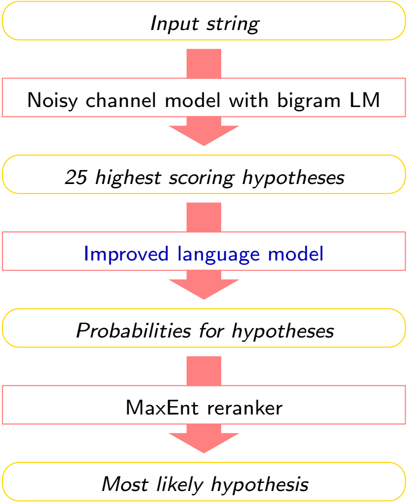
Reranking the Noisy Channel model

- Log probs from source model and channel model are *reranker features*
- MaxEnt reranker can use *additional features* as well

⇒ Best of both noisy channel and machine-learning approaches

- Johnson et al used a parser-based language model

Johnson, Charniak and Lease (2004)

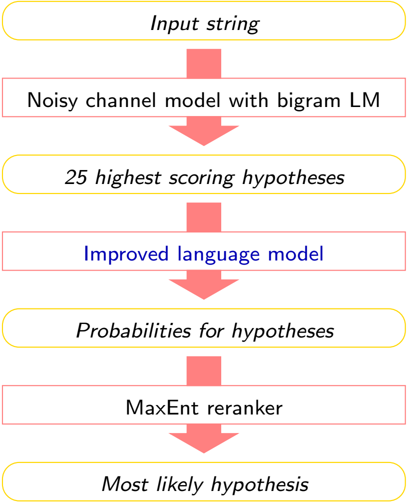


Reranking the Noisy Channel model

- Log probs from source model and channel model are *reranker features*
- MaxEnt reranker can use *additional features* as well

⇒ Best of both noisy channel and machine-learning approaches

- Johnson et al used a parser-based language model



Johnson, Charniak and Lease (2004)

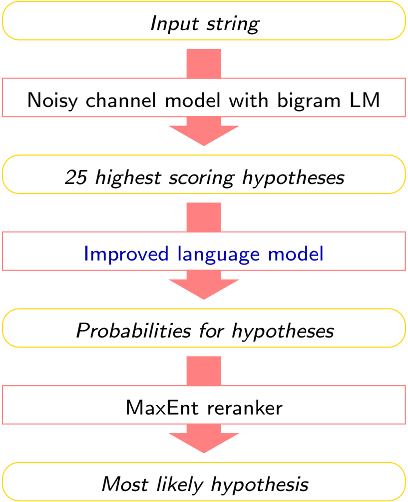
MACQUARIE UNIVERSITY

Reranking the Noisy Channel model

- Log probs from source model and channel model are *reranker features*
- MaxEnt reranker can use *additional features* as well

⇒ Best of both noisy channel and machine-learning approaches

- Johnson et al used a parser-based language model



Johnson, Charniak and Lease (2004)

MACQUARIE UNIVERSITY

Other related work

- Schuler (2010) uses a Hierarchical Hidden Markov Model to simultaneously parse and perform disfluency detection
- Snover (2004) investigates the utility of lexical and prosodic cues for disfluency detection
- Kahn, Lease, Charniak, Johnson and Ostendorf (2005) integrated prosodic cues into the noisy-channel reranker to parse speech-recogniser output
- Zhang, Weng and Feng (2006) uses a MaxEnt model with a large number of features

Other related work

- Schuler (2010) uses a Hierarchical Hidden Markov Model to simultaneously parse and perform disfluency detection
- Snover (2004) investigates the utility of lexical and prosodic cues for disfluency detection
- Kahn, Lease, Charniak, Johnson and Ostendorf (2005) integrated prosodic cues into the noisy-channel reranker to parse speech-recogniser output
- Zhang, Weng and Feng (2006) uses a MaxEnt model with a large number of features

Other related work

- Schuler (2010) uses a Hierarchical Hidden Markov Model to simultaneously parse and perform disfluency detection
- Snover (2004) investigates the utility of lexical and prosodic cues for disfluency detection
- Kahn, Lease, Charniak, Johnson and Ostendorf (2005) integrated prosodic cues into the noisy-channel reranker to parse speech-recogniser output
- Zhang, Weng and Feng (2006) uses a MaxEnt model with a large number of features

Other related work

- Schuler (2010) uses a Hierarchical Hidden Markov Model to simultaneously parse and perform disfluency detection
- Snover (2004) investigates the utility of lexical and prosodic cues for disfluency detection
- Kahn, Lease, Charniak, Johnson and Ostendorf (2005) integrated prosodic cues into the noisy-channel reranker to parse speech-recogniser output
- Zhang, Weng and Feng (2006) uses a MaxEnt model with a large number of features

Outline

Detecting and correcting speech errors in fluent speech

Previous work on disfluency detection

Language models and reranker features

Loss functions

Experimental results

Conclusion

MACQUARIE UNIVERSITY

How does the language model affect performance?

- Is the *size* of the training corpus important?
  - n-gram KN language model trained on *Google Web1T corpus* ($\approx 10^{12}$ words)
- Is it important that the language model is trained on *fluent language*?
  - 4-gram KN language model trained on *Gigaword corpus* (1.6×10^9 words)
- Is it important that the language model is trained on *speech data*?
  - 4-gram KN language model trained on *Fischer corpus* (2.2×10^7 words)
- Is it important that the language model is *disfluency annotated*?
  - 4-gram KN language model trained on *Switchboard corpus* (1.3×10^6 words)

MACQUARIE UNIVERSITY

How does the language model affect performance?

- Is the *size* of the training corpus important?
  - *n*-gram KN language model trained on *Google Web1T corpus* ($\approx 10^{12}$ words)
- Is it important that the language model is trained on *fluent language*?
  - 4-gram KN language model trained on *Gigaword corpus* (1.6×10^9 words)
- Is it important that the language model is trained on *speech data*?
  - 4-gram KN language model trained on *Fischer corpus* (2.2×10^7 words)
- Is it important that the language model is *disfluency annotated*?
  - 4-gram KN language model trained on *Switchboard corpus* (1.3×10^6 words)

MACQUARIE UNIVERSITY

How does the language model affect performance?

- Is the *size* of the training corpus important?
  - *n*-gram KN language model trained on *Google Web1T corpus* ($\approx 10^{12}$ words)
- Is it important that the language model is trained on *fluent language*?
  - 4-gram KN language model trained on *Gigaword corpus* (1.6×10^9 words)
- Is it important that the language model is trained on *speech data*?
  - 4-gram KN language model trained on *Fischer corpus* (2.2×10^7 words)
- Is it important that the language model is *disfluency annotated*?
  - 4-gram KN language model trained on *Switchboard corpus* (1.3×10^6 words)

MACQUARIE UNIVERSITY

How does the language model affect performance?

- Is the *size* of the training corpus important?
  - n-gram KN language model trained on *Google Web1T corpus* ($\approx 10^{12}$ words)
- Is it important that the language model is trained on *fluent language*?
  - 4-gram KN language model trained on *Gigaword corpus* (1.6×10^9 words)
- Is it important that the language model is trained on *speech data*?
  - 4-gram KN language model trained on *Fischer corpus* (2.2×10^7 words)
- Is it important that the language model is *disfluency annotated*?
  - 4-gram KN language model trained on *Switchboard corpus* (1.3×10^6 words)

MACQUARIE UNIVERSITY

How does the language model affect performance?

- Is the *size* of the training corpus important?
  - *n*-gram KN language model trained on *Google Web1T corpus* ($\approx 10^{12}$ words)
- Is it important that the language model is trained on *fluent language*?
  - 4-gram KN language model trained on *Gigaword corpus* (1.6×10^9 words)
- Is it important that the language model is trained on *speech data*?
  - 4-gram KN language model trained on *Fischer corpus* (2.2×10^7 words)
- Is it important that the language model is *disfluency annotated*?
  - 4-gram KN language model trained on *Switchboard corpus* (1.3×10^6 words)

MACQUARIE UNIVERSITY

How does the language model affect performance?

- Is the *size* of the training corpus important?
  - *n*-gram KN language model trained on *Google Web1T corpus* ($\approx 10^{12}$ words)
- Is it important that the language model is trained on *fluent language*?
  - 4-gram KN language model trained on *Gigaword corpus* (1.6×10^9 words)
- Is it important that the language model is trained on *speech data*?
  - 4-gram KN language model trained on *Fischer corpus* (2.2×10^7 words)
- Is it important that the language model is *disfluency annotated*?
  - 4-gram KN language model trained on *Switchboard corpus* (1.3×10^6 words)

MACQUARIE UNIVERSITY

How does the language model affect performance?

- Is the *size* of the training corpus important?
  - n-gram KN language model trained on *Google Web1T corpus* ($\approx 10^{12}$ words)
- Is it important that the language model is trained on *fluent language*?
  - 4-gram KN language model trained on *Gigaword corpus* (1.6×10^9 words)
- Is it important that the language model is trained on *speech data*?
  - 4-gram KN language model trained on *Fischer corpus* (2.2×10^7 words)
- Is it important that the language model is *disfluency annotated*?
  - 4-gram KN language model trained on *Switchboard corpus* (1.3×10^6 words)

MACQUARIE UNIVERSITY

How does the language model affect performance?

- Is the *size* of the training corpus important?
  - *n*-gram KN language model trained on *Google Web1T corpus* ($\approx 10^{12}$ words)
- Is it important that the language model is trained on *fluent language*?
  - 4-gram KN language model trained on *Gigaword corpus* (1.6×10^9 words)
- Is it important that the language model is trained on *speech data*?
  - 4-gram KN language model trained on *Fischer corpus* (2.2×10^7 words)
- Is it important that the language model is *disfluency annotated*?
  - 4-gram KN language model trained on *Switchboard corpus* (1.3×10^6 words)

Additional reranker features

- Bigram language model and channel model *log probabilities from noisy channel model*
- Log probabilities of other language models
- **CopyFlags**: EDITED flags surrounding a sequence of "copied" words (745 features)
- **WordsFlags**: EDITED flags surrounding specific lexical items (256,808 features)
- **SentenceEdgeFlags**: Distance of EDITED flags from the beginning or end of sentence (22 features)

MACQUARIE UNIVERSITY

Additional reranker features

- Bigram language model and channel model *log probabilities from noisy channel model*
- Log probabilities of other language models
- **CopyFlags**: EDITED flags surrounding a sequence of "copied" words (745 features)
- **WordsFlags**: EDITED flags surrounding specific lexical items (256,808 features)
- **SentenceEdgeFlags**: Distance of EDITED flags from the beginning or end of sentence (22 features)

Additional reranker features

- Bigram language model and channel model *log probabilities from noisy channel model*
- Log probabilities of other language models
- **CopyFlags**: EDITED flags surrounding a sequence of "copied" words (745 features)
- **WordsFlags**: EDITED flags surrounding specific lexical items (256,808 features)
- **SentenceEdgeFlags**: Distance of EDITED flags from the beginning or end of sentence (22 features)

Additional reranker features

- Bigram language model and channel model *log probabilities from noisy channel model*
- Log probabilities of other language models
- **CopyFlags**: EDITED flags surrounding a sequence of “copied” words (745 features)
- **WordsFlags**: EDITED flags surrounding specific lexical items (256,808 features)
- **SentenceEdgeFlags**: Distance of EDITED flags from the beginning or end of sentence (22 features)

MACQUARIE UNIVERSITY

Additional reranker features

- Bigram language model and channel model *log probabilities from noisy channel model*
- Log probabilities of other language models
- **CopyFlags**: EDITED flags surrounding a sequence of "copied" words (745 features)
- **WordsFlags**: EDITED flags surrounding specific lexical items (256,808 features)
- **SentenceEdgeFlags**: Distance of EDITED flags from the beginning or end of sentence (22 features)

Outline

Detecting and correcting speech errors in fluent speech

Previous work on disfluency detection

Language models and reranker features

Loss functions

Experimental results

Conclusion

The unbalanced nature of the corpus

$\ldots$ I want a flight $\underbrace{\text{to Boston}}_{\text{Reparandum}}$ $\underbrace{\text{uh I mean}}_{\text{Interregnum}}$ $\underbrace{\text{to Denver}}_{\text{Repair}}$ on Friday $\ldots$

- Switchboard corpus annotates *reparandum, interregnum and repair*
- Trained on Switchboard files sw[23]*.dps (1.3M words)
- Punctuation and partial words ignored
- 31K repairs, average repair length 1.6 words
- Number of training words: *reparandum 50K (3.8%), interregnum 10K (0.8%),* repair 53K (4%), too complicated 24K (1.8%)

The unbalanced nature of the corpus

$\ldots$ I want a flight $\underbrace{\text{to Boston}}_{\text{Reparandum}}$ $\underbrace{\text{uh I mean}}_{\text{Interregnum}}$ $\underbrace{\text{to Denver}}_{\text{Repair}}$ on Friday $\ldots$

- Switchboard corpus annotates *reparandum, interregnum and repair*
- Trained on Switchboard files sw[23]*.dps (1.3M words)
- Punctuation and partial words ignored
- 31K repairs, average repair length 1.6 words
- Number of training words: *reparandum 50K (3.8%), interregnum 10K (0.8%),* repair 53K (4%), too complicated 24K (1.8%)

MACQUARIE UNIVERSITY

The unbalanced nature of the corpus

$\ldots$ I want a flight $\underbrace{\text{to Boston}}_{\text{Reparandum}}$ $\underbrace{\text{uh I mean}}_{\text{Interregnum}}$ $\underbrace{\text{to Denver}}_{\text{Repair}}$ on Friday $\ldots$

- Switchboard corpus annotates *reparandum, interregnum and repair*
- Trained on Switchboard files sw[23]*.dps (1.3M words)
- Punctuation and partial words ignored
- 31K repairs, average repair length 1.6 words
- Number of training words: *reparandum 50K (3.8%), interregnum 10K (0.8%),* repair 53K (4%), too complicated 24K (1.8%)

The unbalanced nature of the corpus

$\ldots$ I want a flight $\underbrace{\text{to Boston}}_{\text{Reparandum}}$ $\underbrace{\text{uh I mean}}_{\text{Interregnum}}$ $\underbrace{\text{to Denver}}_{\text{Repair}}$ on Friday $\ldots$

- Switchboard corpus annotates *reparandum, interregnum and repair*
- Trained on Switchboard files sw[23]*.dps (1.3M words)
- Punctuation and partial words ignored
- 31K repairs, average repair length 1.6 words
- Number of training words: *reparandum 50K (3.8%), interregnum 10K (0.8%),* repair 53K (4%), too complicated 24K (1.8%)

MACQUARIE UNIVERSITY

The unbalanced nature of the corpus

. . . I want a flight $\underbrace{\text{to Boston}}_{\text{Reparandum}}$ $\underbrace{\text{uh I mean}}_{\text{Interregnum}}$ $\underbrace{\text{to Denver}}_{\text{Repair}}$ on Friday . . .

- Switchboard corpus annotates *reparandum, interregnum and repair*
- Trained on Switchboard files sw[23]*.dps (1.3M words)
- Punctuation and partial words ignored
- 31K repairs, average repair length 1.6 words
- Number of training words: *reparandum 50K (3.8%), interregnum 10K (0.8%),* repair 53K (4%), too complicated 24K (1.8%)

Evaluate using f-score instead of accuracy

- Only around 5% words are EDITED ⇒ trivial classifier that always predicts NOTEDITED scores 95% accuracy
- *F-score* f is geometric mean of precision and recall

$$f = \frac{2c}{g+e}$$

  where g and e are number of gold and predicted EDITED words, and c is the number of correct EDITED words
- Trivial classifier has 100% precision but 0% recall ⇒ f-score = 0
- Alternative measure: *error rate* (= number of EDITED word errors divided by number of true EDITED words)

Charniak and Johnson (2001), Rich Text Evaluation (2004)

MACQUARIE UNIVERSITY

Evaluate using f-score instead of accuracy

- Only around 5% words are EDITED $\Rightarrow$ trivial classifier that always predicts NOTEDITED scores 95% accuracy
- *F-score* f is geometric mean of precision and recall

$$f = \frac{2c}{g + e}$$

  where g and e are number of gold and predicted EDITED words, and c is the number of correct EDITED words
- Trivial classifier has 100% precision but 0% recall $\Rightarrow$ f-score $= 0$
- Alternative measure: *error rate* ($=$ number of EDITED word errors divided by number of true EDITED words)

Charniak and Johnson (2001), Rich Text Evaluation (2004)

MACQUARIE UNIVERSITY

Evaluate using f-score instead of accuracy

- Only around 5% words are EDITED $\Rightarrow$ trivial classifier that always predicts NOTEDITED scores 95% accuracy
- *F-score* f is geometric mean of precision and recall

$$f = \frac{2c}{g+e}$$

  where g and e are number of gold and predicted EDITED words, and c is the number of correct EDITED words
- Trivial classifier has 100% precision but 0% recall $\Rightarrow$ f-score $= 0$
- Alternative measure: *error rate* ($=$ number of EDITED word errors divided by number of true EDITED words)

Charniak and Johnson (2001), Rich Text Evaluation (2004)

Evaluate using f-score instead of accuracy

- Only around 5% words are EDITED $\Rightarrow$ trivial classifier that always predicts NOTEDITED scores 95% accuracy
- *F-score* f is geometric mean of precision and recall

$$f = \frac{2c}{g+e}$$

  where g and e are number of gold and predicted EDITED words, and c is the number of correct EDITED words
- Trivial classifier has 100% precision but 0% recall $\Rightarrow$ f-score $= 0$
- Alternative measure: *error rate* ($=$ number of EDITED word errors divided by number of true EDITED words)

Charniak and Johnson (2001), Rich Text Evaluation (2004)

Modify classifier to optimise expected f-score

- A standard MaxEnt estimator optimises *log-loss*, which weights EDITED ∼ NOTEDITED errors equally
- We can modify the estimator so it optimises an *approximate expected f-score* instead

$$\tilde{f} = \frac{2\mathrm{E}_w[c]}{g + \mathrm{E}_w[e]}$$

  - approximation assumes that *expectation distributes over division*
- $\tilde{f}$ and its derivatives can be easily calculated

⇒ use L-BFGS to estimate feature weights $\widehat{w}$

- Similar calculation can be done for *expected error rate*

Jansche (2005), Smith and Eisner (2006)

MACQUARIE UNIVERSITY

Modify classifier to optimise expected f-score

- A standard MaxEnt estimator optimises *log-loss*, which weights EDITED $\sim$ NOTEDITED errors equally
- We can modify the estimator so it optimises an *approximate expected f-score* instead

$$\tilde{f} = \frac{2\mathrm{E}_{\boldsymbol{w}}[c]}{g + \mathrm{E}_{\boldsymbol{w}}[e]}$$

  - approximation assumes that *expectation distributes over division*
- $\tilde{f}$ and its derivatives can be easily calculated

$\Rightarrow$ use L-BFGS to estimate feature weights $\widehat{w}$

- Similar calculation can be done for *expected error rate*

Jansche (2005), Smith and Eisner (2006)

MACQUARIE UNIVERSITY

Modify classifier to optimise expected f-score

- A standard MaxEnt estimator optimises *log-loss*, which weights $\textsc{Edited} \sim \textsc{NotEdited}$ errors equally
- We can modify the estimator so it optimises an *approximate expected f-score* instead

$$\tilde{f} \;=\; \frac{2\mathrm{E}_{\boldsymbol{w}}[c]}{g + \mathrm{E}_{\boldsymbol{w}}[e]}$$

  - approximation assumes that *expectation distributes over division*
- $\tilde{f}$ and its derivatives can be easily calculated

$\Rightarrow$ use L-BFGS to estimate feature weights $\widehat{w}$

- Similar calculation can be done for *expected error rate*

Jansche (2005), Smith and Eisner (2006)

MACQUARIE UNIVERSITY

Modify classifier to optimise expected f-score

- A standard MaxEnt estimator optimises *log-loss*, which weights EDITED ~ NOTEDITED errors equally
- We can modify the estimator so it optimises an *approximate expected f-score* instead

$$\tilde{f} \quad = \quad \frac{2\mathrm{E}_{\boldsymbol{w}}[c]}{g + \mathrm{E}_{\boldsymbol{w}}[e]}$$

  - approximation assumes that *expectation distributes over division*
- $\tilde{f}$ and its derivatives can be easily calculated

⇒ use L-BFGS to estimate feature weights $\widehat{w}$

- Similar calculation can be done for *expected error rate*

Jansche (2005), Smith and Eisner (2006)

Modify classifier to optimise expected f-score

- A standard MaxEnt estimator optimises *log-loss*, which weights EDITED $\sim$ NOTEDITED errors equally
- We can modify the estimator so it optimises an *approximate expected f-score* instead

$$\tilde{f} \;=\; \frac{2\mathrm{E}_{\boldsymbol{w}}[c]}{g + \mathrm{E}_{\boldsymbol{w}}[e]}$$

  - approximation assumes that *expectation distributes over division*
- $\tilde{f}$ and its derivatives can be easily calculated

$\Rightarrow$ use L-BFGS to estimate feature weights $\widehat{w}$

- Similar calculation can be done for *expected error rate*

Jansche (2005), Smith and Eisner (2006)

MACQUARIE UNIVERSITY

Modify classifier to optimise expected f-score

- A standard MaxEnt estimator optimises *log-loss*, which weights EDITED $\sim$ NOTEDITED errors equally
- We can modify the estimator so it optimises an *approximate expected f-score* instead

$$\tilde{f} \quad = \quad \frac{2\mathrm{E}_{\boldsymbol{w}}[c]}{g + \mathrm{E}_{\boldsymbol{w}}[e]}$$

  - approximation assumes that *expectation distributes over division*
- $\tilde{f}$ and its derivatives can be easily calculated

$\Rightarrow$ use L-BFGS to estimate feature weights $\widehat{w}$

- Similar calculation can be done for *expected error rate*

Jansche (2005), Smith and Eisner (2006)

MACQUARIE UNIVERSITY

Outline

Detecting and correcting speech errors in fluent speech

Previous work on disfluency detection

Language models and reranker features

Loss functions

Experimental results

Conclusion

Experimental set-up

- All partial words and punctuation were deleted from training, held-out and test
- Training data: Switchboard sw[23]$\star$.dps files
- Held-out data: Switchboard sw4[5-9]$\star$.dps files
- Test data (only used once): Switchboard sw[0-1]$\star$.dps files

Johnson and Charniak (2001)

Experimental set-up

- All partial words and punctuation were deleted from training, held-out and test
- Training data: Switchboard sw[23]$\star$.dps files
- Held-out data: Switchboard sw4[5-9]$\star$.dps files
- Test data (only used once): Switchboard sw[0-1]$\star$.dps files

Johnson and Charniak (2001)

Experimental set-up

- All partial words and punctuation were deleted from training, held-out and test
- Training data: Switchboard sw[23]$\star$.dps files
- Held-out data: Switchboard sw4[5-9]$\star$.dps files
- Test data (only used once): Switchboard sw[0-1]$\star$.dps files

Johnson and Charniak (2001)

Experimental set-up

- All partial words and punctuation were deleted from training, held-out and test
- Training data: Switchboard sw[23]$\star$.dps files
- Held-out data: Switchboard sw4[5-9]$\star$.dps files
- Test data (only used once): Switchboard sw[0-1]$\star$.dps files

Johnson and Charniak (2001)

Results on held-out data

Model	F-score	
NC (noisy channel, no reranking)	0.756	
Model	**log loss**	**expected f-score loss**
NC + Switchboard	0.776	0.791
NC + Fisher	0.771	0.797
NC + Gigaword	0.777	0.797
NC + Web1T	0.781	0.798
NC + Reranker Feat.	0.824	0.827
NC + Reranker Feat. + Switchboard	0.827	0.828
NC + Reranker Feat. + Fisher	0.841	0.856
NC + Reranker Feat. + Gigaword	0.843	0.852
NC + Reranker Feat. + Web1T	0.843	0.850
NC + Reranker Feat. + All LM	0.841	**0.857**

MACQUARIE UNIVERSITY

Results on test corpus

- One run on test corpus, NC + Reranker Feat. + All LM, expected f-score loss: **0.838**
- Previous results:
  - Charniak and Johnson (2001) (Boosting classifier): 0.759
  - Johnson and Charniak (2004) (Noisy channel model): 0.797
  - Zhang, Weng and Feng (2006) (Ultra-large feature space, prosodic features): 0.8205 (note: test data not exactly comparable)

MACQUARIE UNIVERSITY

Results on test corpus

- One run on test corpus, NC + Reranker Feat. + All LM, expected f-score loss: **0.838**
- Previous results:
  - Charniak and Johnson (2001) (Boosting classifier): 0.759
  - Johnson and Charniak (2004) (Noisy channel model): 0.797
  - Zhang, Weng and Feng (2006) (Ultra-large feature space, prosodic features): 0.8205 (note: test data not exactly comparable)

MACQUARIE UNIVERSITY

Results on test corpus

- One run on test corpus, NC + Reranker Feat. + All LM, expected f-score loss: **0.838**
- Previous results:
  - Charniak and Johnson (2001) (Boosting classifier): 0.759
  - Johnson and Charniak (2004) (Noisy channel model): 0.797
  - Zhang, Weng and Feng (2006) (Ultra-large feature space, prosodic features): 0.8205 (note: test data not exactly comparable)

MACQUARIE UNIVERSITY

Results on test corpus

- One run on test corpus, NC + Reranker Feat. + All LM, expected f-score loss: **0.838**
- Previous results:
  - Charniak and Johnson (2001) (Boosting classifier): 0.759
  - Johnson and Charniak (2004) (Noisy channel model): 0.797
  - Zhang, Weng and Feng (2006) (Ultra-large feature space, prosodic features): 0.8205 (note: test data not exactly comparable)

MACQUARIE UNIVERSITY

Results on test corpus

- One run on test corpus, NC + Reranker Feat. + All LM, expected f-score loss: **0.838**
- Previous results:
  - Charniak and Johnson (2001) (Boosting classifier): 0.759
  - Johnson and Charniak (2004) (Noisy channel model): 0.797
  - Zhang, Weng and Feng (2006) (Ultra-large feature space, prosodic features): 0.8205 (note: test data not exactly comparable)

MACQUARIE UNIVERSITY

Outline

Detecting and correcting speech errors in fluent speech

Previous work on disfluency detection

Language models and reranker features

Loss functions

Experimental results

Conclusion

Conclusions

- The noisy channel model is useful for detecting speech disfluencies
- A reranker can markedly improve performance
- The choice of training data used in the language model does not seem to be very important
  - not necessary for LM to be trained on disfluency-annotated data
  - not necessary for LM to be trained on speech data
- Using additional reranker features boosts performance still further
- Optimising a loss function more closely related to the evaluation metric further boosts performance

MACQUARIE UNIVERSITY

Conclusions

- The noisy channel model is useful for detecting speech disfluencies
- A reranker can markedly improve performance
- The choice of training data used in the language model does not seem to be very important
  - not necessary for LM to be trained on disfluency-annotated data
  - not necessary for LM to be trained on speech data
- Using additional reranker features boosts performance still further
- Optimising a loss function more closely related to the evaluation metric further boosts performance

MACQUARIE UNIVERSITY

Conclusions

- The noisy channel model is useful for detecting speech disfluencies
- A reranker can markedly improve performance
- The choice of training data used in the language model does not seem to be very important
  - not necessary for LM to be trained on disfluency-annotated data
  - not necessary for LM to be trained on speech data
- Using additional reranker features boosts performance still further
- Optimising a loss function more closely related to the evaluation metric further boosts performance

MACQUARIE UNIVERSITY

Conclusions

- The noisy channel model is useful for detecting speech disfluencies
- A reranker can markedly improve performance
- The choice of training data used in the language model does not seem to be very important
  - not necessary for LM to be trained on disfluency-annotated data
  - not necessary for LM to be trained on speech data
- Using additional reranker features boosts performance still further
- Optimising a loss function more closely related to the evaluation metric further boosts performance

MACQUARIE UNIVERSITY

Conclusions

- The noisy channel model is useful for detecting speech disfluencies
- A reranker can markedly improve performance
- The choice of training data used in the language model does not seem to be very important
  - not necessary for LM to be trained on disfluency-annotated data
  - not necessary for LM to be trained on speech data
- Using additional reranker features boosts performance still further
- Optimising a loss function more closely related to the evaluation metric further boosts performance

MACQUARIE UNIVERSITY

Conclusions

- The noisy channel model is useful for detecting speech disfluencies
- A reranker can markedly improve performance
- The choice of training data used in the language model does not seem to be very important
  - not necessary for LM to be trained on disfluency-annotated data
  - not necessary for LM to be trained on speech data
- Using additional reranker features boosts performance still further
- Optimising a loss function more closely related to the evaluation metric further boosts performance

Conclusions

- The noisy channel model is useful for detecting speech disfluencies
- A reranker can markedly improve performance
- The choice of training data used in the language model does not seem to be very important
    - not necessary for LM to be trained on disfluency-annotated data
    - not necessary for LM to be trained on speech data
- Using additional reranker features boosts performance still further
- Optimising a loss function more closely related to the evaluation metric further boosts performance

Future work

- Work with real speech recogniser output (Kahn et al, 2005)
- Experiment with a parsing-based language model trained on large (unlabelled) corpus
- Develop a system that does not require sentence-segmented input
  $\Rightarrow$ *incremental parser-based language model* trained from semi-supervised data

MACQUARIE UNIVERSITY

Future work

- Work with real speech recogniser output (Kahn et al, 2005)
- Experiment with a parsing-based language model trained on large (unlabelled) corpus
- Develop a system that does not require sentence-segmented input
  - $\Rightarrow$ *incremental parser-based language model* trained from semi-supervised data

MACQUARIE UNIVERSITY

Future work

- Work with real speech recogniser output (Kahn et al, 2005)
- Experiment with a parsing-based language model trained on large (unlabelled) corpus
- Develop a system that does not require sentence-segmented input

  ⇒ *incremental parser-based language model* trained from semi-supervised data

MACQUARIE UNIVERSITY

Future work

- Work with real speech recogniser output (Kahn et al, 2005)
- Experiment with a parsing-based language model trained on large (unlabelled) corpus
- Develop a system that does not require sentence-segmented input
  - $\Rightarrow$ *incremental parser-based language model* trained from semi-supervised data

MACQUARIE UNIVERSITY

Acknowledgements

- Australian Research Council Discovery Project DP110102593
- Australian Research Council's "Thinking Head Project"
- ARC/NHMRC Special Research Initiative Grant TS0669874

MACQUARIE UNIVERSITY