Sequence-to-Sequence Natural Language Generation

Ondřej Dušek

Interaction Lab
MACS, Heriot Watt University, Edinburgh

work done with Filip Jurčíček
at Charles University in Prague

November 10, 2016
DILiGENt project meeting, Edinburgh

Outline of this Talk

1. Introduction to the problem
    - our task + problems we are solving

Outline of this Talk

1. Introduction to the problem
   - our task + problems we are solving
2. Sequence-to-sequence generation
   a) model architecture
   b) experiments on the BAGEL set

Outline of this Talk

1. Introduction to the problem
   - our task + problems we are solving
2. Sequence-to-sequence generation
   a) model architecture
   b) experiments on the BAGEL set
3. Context-aware extensions
   a) making the basic seq2seq setup context-aware
   b) experiments on our public transport dataset

Outline of this Talk

1. Introduction to the problem
   - our task + problems we are solving
2. Sequence-to-sequence generation
   a) model architecture
   b) experiments on the BAGEL set
3. Context-aware extensions
   a) making the basic seq2seq setup context-aware
   b) experiments on our public transport dataset
4. Future work ideas

NLG in Spoken Dialogue Systems

- converting a meaning representation (dialogue acts, DAs) to a sentence

inform(name=X,eattype=restaurant,food=Italian,area=riverside)

↓

X is an Italian restaurant near the river.

NLG in Spoken Dialogue Systems

- converting a meaning representation (dialogue acts, DAs) to a sentence

inform(name=X,eattype=restaurant,food=Italian,area=riverside)

↓

X is an Italian restaurant near the river.

- no content selection in SDS

NLG in Spoken Dialogue Systems

- converting a meaning representation (dialogue acts, DAs) to a sentence

inform(name=X,eattype=restaurant,food=Italian,area=riverside)

↓

X is an Italian restaurant near the river.

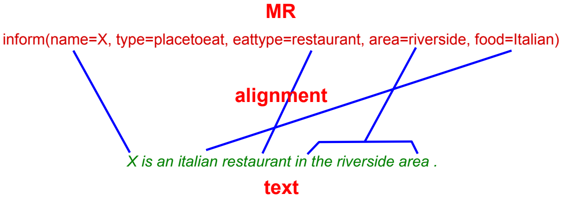
  - no content selection in SDS
- input: from dialogue manager
- output: to TTS

Generating from Unaligned Data

- earlier, NLG systems required:
  a) manual alignments
  b) alignment preprocessing step

Generating from Unaligned Data

- earlier, NLG systems required:
  a) manual alignments
  b) alignment preprocessing step



Generating from Unaligned Data

- earlier, NLG systems required:
  a) manual alignments
  b) alignment preprocessing step
- we learn alignments jointly

MR

inform(name=X, type=placetoeat, eattype=restaurant, area=riverside, food=Italian)

X is an italian restaurant in the riverside area .

text

Generating from Unaligned Data

- earlier, NLG systems required:
  a) manual alignments
  b) alignment preprocessing step
- we learn alignments jointly
  - no error acummulation / manual annotation
  - alignment is latent (needs not be hard/1:1)

MR

inform(name=X, type=placetoeat, eattype=restaurant, area=riverside, food=Italian)

X is an italian restaurant in the riverside area .

text

Generating from Unaligned Data

- earlier, NLG systems required:
  a) manual alignments
  b) alignment preprocessing step
- we learn alignments jointly
  - no error acummulation / manual annotation
  - alignment is latent (needs not be hard/1:1)

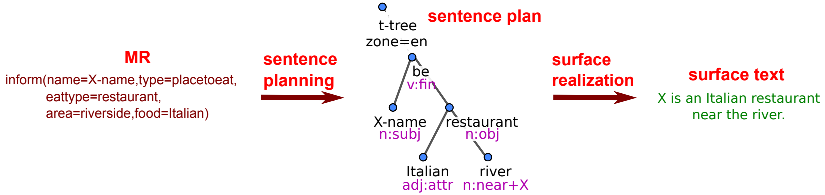
inform(name=X-name, type=placetoeat, **area=centre**, eattype=restaurant, near=X-near)
*The X restaurant is **conveniently** located near X, **right in the city center**.*

inform(name=X-name, type=placetoeat, **foodtype=Chinese_takeaway**)
*X serves **Chinese food** and has a **takeaway** possibility.*

inform(name=X-name, type=placetoeat, **pricerange=cheap**)
***Prices** at X are **quite cheap**.*

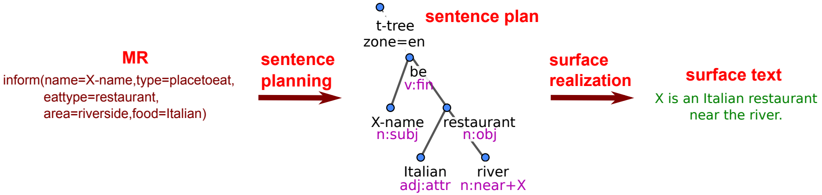
Two-Step and Joint NLG Setups

- NLG pipeline traditionally divided into:
  1. sentence planning – decide on the overall sentence structure
  2. surface realization – decide on specific word forms, linearize



Two-Step and Joint NLG Setups

- NLG pipeline traditionally divided into:
  1. sentence planning – decide on the overall sentence structure
  2. surface realization – decide on specific word forms, linearize
- some NLG systems join this into a single step



Two-Step and Joint NLG Setups

- NLG pipeline traditionally divided into:
  1. sentence planning – decide on the overall sentence structure
  2. surface realization – decide on specific word forms, linearize
- some NLG systems join this into a single step

Two-Step and Joint NLG Setups

- NLG pipeline traditionally divided into:
  1. sentence planning – decide on the overall sentence structure
  2. surface realization – decide on specific word forms, linearize
- some NLG systems join this into a single step
  - two-step setup simplifies structure generation by abstracting away from surface grammar

Two-Step and Joint NLG Setups

- NLG pipeline traditionally divided into:
    1. sentence planning – decide on the overall sentence structure
    2. surface realization – decide on specific word forms, linearize
- some NLG systems join this into a single step
    - two-step setup simplifies structure generation by abstracting away from surface grammar
    - joint setup avoids error accumulation over a pipeline

Two-Step and Joint NLG Setups

- NLG pipeline traditionally divided into:
  1. sentence planning – decide on the overall sentence structure
  2. surface realization – decide on specific word forms, linearize
- some NLG systems join this into a single step
  - two-step setup simplifies structure generation by abstracting away from surface grammar
  - joint setup avoids error accumulation over a pipeline
- we can do both in one system

Entrainment in Dialogues and NLG

- speakers are influenced by previous utterances
  - adapting (entraining) to each other
  - reusing lexicon and syntax

Entrainment in Dialogues and NLG

- speakers are influenced by previous utterances
  - adapting (entraining) to each other
  - reusing lexicon and syntax

how bout the next ride

Sorry, I did not find a later option.

I'm sorry, the next ride was not found.

Entrainment in Dialogues and NLG

- speakers are influenced by previous utterances
  - adapting (entraining) to each other
  - reusing lexicon and syntax
- entrainment is natural, subconscious, helps conversation success

Entrainment in Dialogues and NLG

- speakers are influenced by previous utterances
  - adapting (entraining) to each other
  - reusing lexicon and syntax
- entrainment is natural, subconscious, helps conversation success
- natural source of variation

Entrainment in Dialogues and NLG

- speakers are influenced by previous utterances
  - adapting (entraining) to each other
  - reusing lexicon and syntax
- entrainment is natural, subconscious, helps conversation success
- natural source of variation
- typical NLG only takes the input DA into account

Entrainment in Dialogues and NLG

- speakers are influenced by previous utterances
  - adapting (entraining) to each other
  - reusing lexicon and syntax
- entrainment is natural, subconscious, helps conversation success
- natural source of variation
- typical NLG only takes the input DA into account
  - no way of adapting to user's way of speaking

Entrainment in Dialogues and NLG

- speakers are influenced by previous utterances
  - adapting (entraining) to each other
  - reusing lexicon and syntax
- entrainment is natural, subconscious, helps conversation success
- natural source of variation
- typical NLG only takes the input DA into account
  - no way of adapting to user's way of speaking
  - no output variance (must be fabricated, e.g., by sampling)

Entrainment in Dialogues and NLG

- speakers are influenced by previous utterances
  - adapting (entraining) to each other
  - reusing lexicon and syntax
- entrainment is natural, subconscious, helps conversation success
- natural source of variation
- typical NLG only takes the input DA into account
  - no way of adapting to user's way of speaking
  - no output variance (must be fabricated, e.g., by sampling)
- entrainment in NLG limited to rule-based systems so far

Entrainment in Dialogues and NLG

- speakers are influenced by previous utterances
  - adapting (entraining) to each other
  - reusing lexicon and syntax
- entrainment is natural, subconscious, helps conversation success
- natural source of variation
- typical NLG only takes the input DA into account
  - no way of adapting to user's way of speaking
  - no output variance (must be fabricated, e.g., by sampling)
- entrainment in NLG limited to rule-based systems so far
- our system is trainable and entrains/adapts

Our NLG system

- based on sequence-to-sequence neural network models

Our NLG system

- based on sequence-to-sequence neural network models
- ✓ trainable from unaligned pairs of input DAs + sentences

Our NLG system

- based on sequence-to-sequence neural network models
- ✓ trainable from unaligned pairs of input DAs + sentences
- ✓ two operating modes
  - we can compare 2-step and joint setups in a single architecture

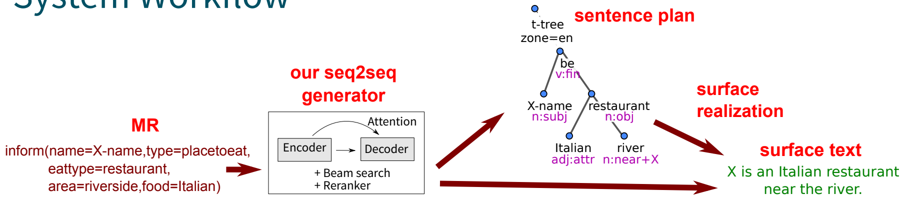
Our NLG system

- based on sequence-to-sequence neural network models
- ✓ trainable from unaligned pairs of input DAs + sentences
- ✓ two operating modes
  - we can compare 2-step and joint setups in a single architecture
- ✓ learns to produce meaningful outputs from very little training data

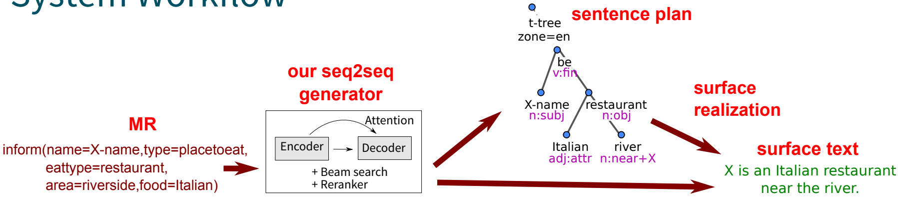
Our NLG system

- based on sequence-to-sequence neural network models
- ✓ trainable from unaligned pairs of input DAs + sentences
- ✓ two operating modes
  - we can compare 2-step and joint setups in a single architecture
- ✓ learns to produce meaningful outputs from very little training data
- ✓ context-aware: adapts to previous user utterance

System Workflow

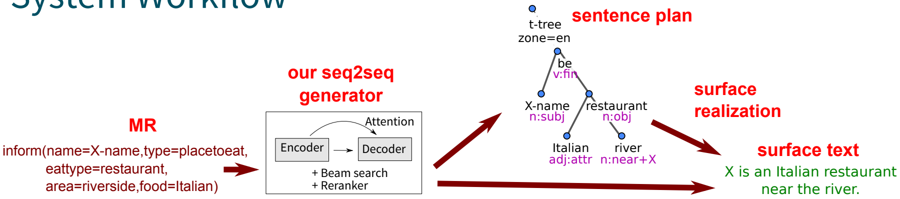


System Workflow



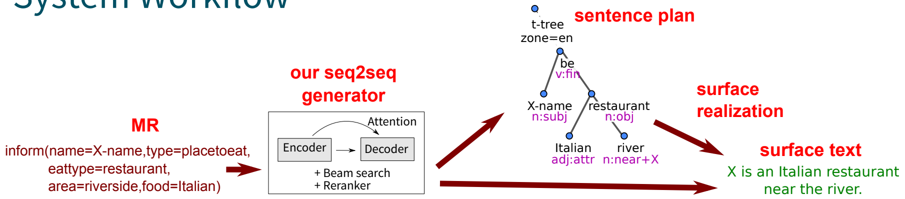
- main generator based on sequence-to-sequence NNs

System Workflow



- main generator based on sequence-to-sequence NNs
- input: tokenized DAs

System Workflow

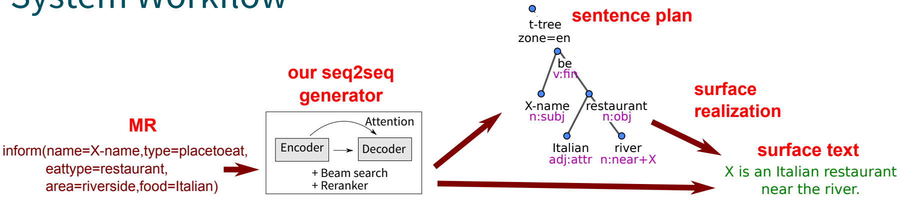


- main generator based on sequence-to-sequence NNs
- input: tokenized DAs
- output:

  2-step mode – deep syntax trees, in bracketed format

(<root> <root> ((X-name n:subj) be v:fin ((Italian adj:attr) restaurant n:obj (river n:near+X))))

System Workflow

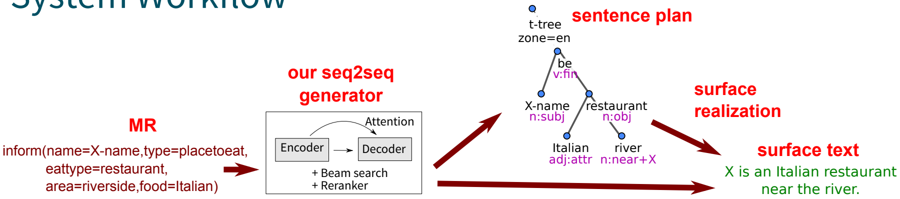


- main generator based on sequence-to-sequence NNs
- input: tokenized DAs
- output:

  2-step mode – deep syntax trees, in bracketed format

  joint mode – sentences

System Workflow

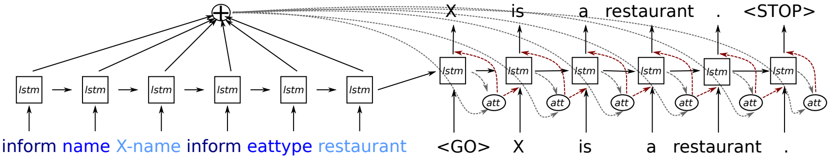


- main generator based on sequence-to-sequence NNs
- input: tokenized DAs
- output:

  2-step mode – deep syntax trees, in bracketed format

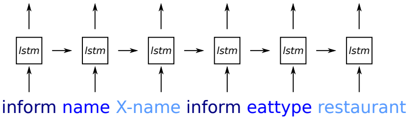
  joint mode – sentences
- 2-step mode: deep syntax trees post-processed by a surface realizer

Our Seq2seq Generator architecture



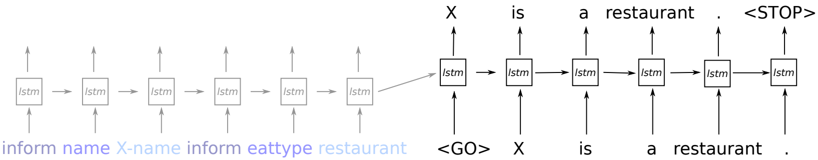
- Sequence-to-sequence models with attention

Our Seq2seq Generator architecture



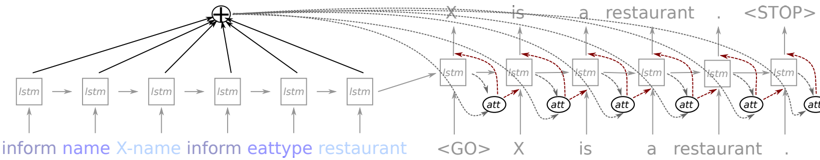
- Sequence-to-sequence models with attention
  - Encoder LSTM RNN: encode DA into hidden states

Our Seq2seq Generator architecture



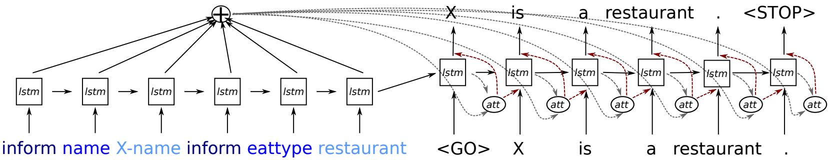
- Sequence-to-sequence models with attention
  - Encoder LSTM RNN: encode DA into hidden states
  - Decoder LSTM RNN: generate output tokens

Our Seq2seq Generator architecture



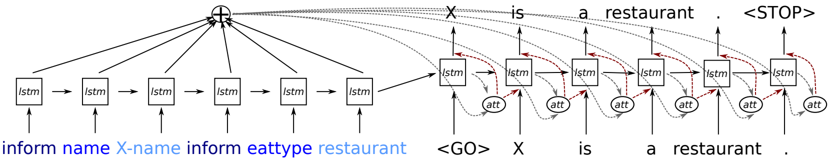
- Sequence-to-sequence models with attention
  - Encoder LSTM RNN: encode DA into hidden states
  - Decoder LSTM RNN: generate output tokens
  - attention model: weighing encoder hidden states

Our Seq2seq Generator architecture



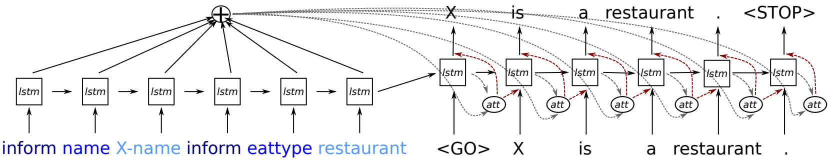
- Sequence-to-sequence models with attention
  - Encoder LSTM RNN: encode DA into hidden states
  - Decoder LSTM RNN: generate output tokens
  - attention model: weighing encoder hidden states
- basic greedy generation

Our Seq2seq Generator architecture



- Sequence-to-sequence models with attention
  - Encoder LSTM RNN: encode DA into hidden states
  - Decoder LSTM RNN: generate output tokens
  - attention model: weighing encoder hidden states
- basic greedy generation
  + beam search, *n*-best list outputs

Our Seq2seq Generator architecture



- Sequence-to-sequence models with attention
  - Encoder LSTM RNN: encode DA into hidden states
  - Decoder LSTM RNN: generate output tokens
  - attention model: weighing encoder hidden states
- basic greedy generation
  + beam search, *n*-best list outputs
  + reranker ($\rightarrow$)

Reranker

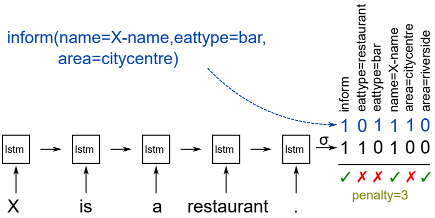
- generator may not cover the input DA perfectly
  - missing / superfluous information

Reranker

- generator may not cover the input DA perfectly
  - missing / superfluous information
  - we would like to penalize such cases

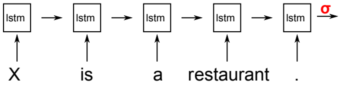
Reranker

- generator may not cover the input DA perfectly
  - missing / superfluous information
  - we would like to penalize such cases
- check whether output conforms to the input DA + rerank



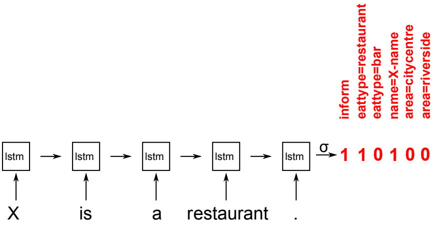
Reranker

- generator may not cover the input DA perfectly
  - missing / superfluous information
  - we would like to penalize such cases
- check whether output conforms to the input DA + rerank
  - NN with LSTM encoder + sigmoid classification layer



Reranker

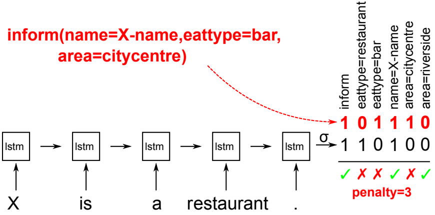
- generator may not cover the input DA perfectly
  - missing / superfluous information
  - we would like to penalize such cases
- check whether output conforms to the input DA + rerank
  - NN with LSTM encoder + sigmoid classification layer



- 1-hot DA representation

Reranker

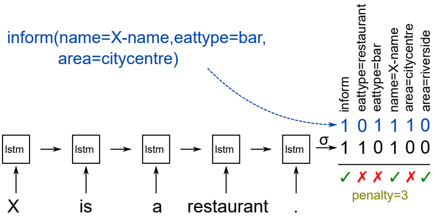
- generator may not cover the input DA perfectly
  - missing / superfluous information
  - we would like to penalize such cases
- check whether output conforms to the input DA + rerank
  - NN with LSTM encoder + sigmoid classification layer



- 1-hot DA representation
- penalty = Hamming distance from input DA (on 1-hot vectors)

Reranker

- generator may not cover the input DA perfectly
  - missing / superfluous information
  - we would like to penalize such cases
- check whether output conforms to the input DA + rerank
  - NN with LSTM encoder + sigmoid classification layer



- 1-hot DA representation
- penalty = Hamming distance from input DA (on 1-hot vectors)

Experiments

- BAGEL dataset:
  202 DAs / 404 sentences, restaurant information

Experiments

- BAGEL dataset:
  202 DAs / 404 sentences, restaurant information
    - much less data than previous seq2seq methods

Experiments

- BAGEL dataset:
  202 DAs / 404 sentences, restaurant information
  - much less data than previous seq2seq methods
  - partially delexicalized (names, phone numbers $\rightarrow$ “X”)

Experiments

- BAGEL dataset:
  202 DAs / 404 sentences, restaurant information
  - much less data than previous seq2seq methods
  - partially delexicalized (names, phone numbers → "X")
  - manual alignment provided, but we do not use it

Experiments

- BAGEL dataset:
  202 DAs / 404 sentences, restaurant information
  - much less data than previous seq2seq methods
  - partially delexicalized (names, phone numbers $\rightarrow$ "X")
  - manual alignment provided, but we do not use it
- 10-fold cross-validation
  - automatic metrics: BLEU, NIST

Experiments

- BAGEL dataset:
  202 DAs / 404 sentences, restaurant information
    - much less data than previous seq2seq methods
    - partially delexicalized (names, phone numbers $\rightarrow$ “X”)
    - manual alignment provided, but we do not use it
- 10-fold cross-validation
    - automatic metrics: BLEU, NIST
    - manual evaluation: semantic errors on 20% data (missing/irrelevant/repeated)

Results

prev

Setup	BLEU	NIST	ERR
Mairesse et al. (2010) *– alignments*	∼67	-	0
Dušek & Jurčíček (2015)	59.89	5.231	30

Results

		Setup	BLEU	NIST	ERR
prev		Mairesse et al. (2010) – *alignments*	∼67	-	0
		Dušek & Jurčíček (2015)	59.89	5.231	30
our	*two-step*	Greedy with trees	55.29	5.144	20
		+ Beam search (beam size 100)	58.59	5.293	28
		+ Reranker (beam size 5)	60.77	5.487	24
		(beam size 10)	60.93	5.510	25
		(beam size 100)	60.44	5.514	**19**
	joint	Greedy into strings	52.54	5.052	37
		+ Beam search (beam size 100)	55.84	5.228	32
		+ Reranker (beam size 5)	61.18	5.507	27
		(beam size 10)	62.40	5.614	21
		(beam size 100)	**62.76**	**5.669**	**19**

Sample Outputs

Input DA	inform(name=X-name, type=placetoeat, eattype=restaurant, area=citycentre, near=X-near, food="Chinese takeaway", food=Japanese)
Reference	X is a Chinese takeaway and Japanese restaurant in the city centre near X.
Greedy with trees	X is a restaurant offering chinese takeaway in the centre of town near X. [Japanese]
+ Beam search	X is a restaurant and japanese food and chinese takeaway.
+ Reranker	X is a restaurant serving japanese food in the centre of the city that offers chinese takeaway.
Greedy into strings	X is a restaurant offering italian and indian takeaway in the city centre area near X. [Japanese, Chinese]
+ Beam search	X is a restaurant that serves fusion chinese takeaway in the riverside area near X. [Japanese, citycentre]
+ Reranker	X is a japanese restaurant in the city centre near X providing chinese food. [takeaway]

Sample Outputs

Input DA	inform(name=X-name, type=placetoeat, eattype=restaurant, area=riverside, food=French)
Reference	X is a French restaurant on the riverside.
Greedy with trees	X is a restaurant providing french and continental and by the river.
+ Beam search	X is a restaurant that serves french takeaway. [riverside]
+ Reranker	X is a french restaurant in the riverside area.
Greedy into strings	X is a restaurant in the riverside that serves italian food. [French]
+ Beam search	X is a restaurant in the riverside that serves italian food. [French]
+ Reranker	X is a restaurant in the riverside area that serves french food.

Sample Outputs

Input DA	inform(name=X-name, type=placetoeat, eattype=restaurant, near=X-near, food=Continental, food=French)
Reference	X is a French and Continental restaurant near X.
Greedy with trees	X is a french restaurant that serves french food and near X. [Continental]
+ Beam search	X is a french restaurant that serves french food and near X. [Continental]
+ Reranker	X is a restaurant serving french and continental food near X.
Greedy into strings	X is a french and continental style restaurant near X.
+ Beam search	X is a french and continental style restaurant near X.
+ Reranker	X is a restaurant providing french and continental food, near X.

Conclusions

- both setups produce mostly valid outputs despite limited training data
  - correct domain style
  - mostly fluent

Conclusions

- both setups produce mostly valid outputs despite limited training data
  - correct domain style
  - mostly fluent
- different types of errors
  - joint: confusion of similar items (*Italian* vs. *French*)
  - 2-step: disfluency, missing/superfluous/repeated items

Conclusions

- both setups produce mostly valid outputs despite limited training data
  - correct domain style
  - mostly fluent
- different types of errors
  - joint: confusion of similar items (*Italian* vs. *French*)
  - 2-step: disfluency, missing/superfluous/repeated items
- joint generation works better on our domain (+2% BLEU)

Conclusions

- both setups produce mostly valid outputs despite limited training data
  - correct domain style
  - mostly fluent
- different types of errors
  - joint: confusion of similar items (*Italian* vs. *French*)
  - 2-step: disfluency, missing/superfluous/repeated items
- joint generation works better on our domain (+2% BLEU)
- better results than our previous work with unaligned data

Adding Entrainment to Trainable NLG

- Aim: condition generation on preceding context

Adding Entrainment to Trainable NLG

- Aim: condition generation on preceding context
- Problem: data sparsity

Adding Entrainment to Trainable NLG

- Aim: condition generation on preceding context
- Problem: data sparsity
- Solution: Limit context to just preceding user utterance
  - likely to have strongest entrainment impact

Adding Entrainment to Trainable NLG

- Aim: condition generation on preceding context
- Problem: data sparsity
- Solution: Limit context to just preceding user utterance
  - likely to have strongest entrainment impact
- Need for context-aware training data: we collected a new set
  - input DA
  - natural language sentence(s)

inform(from_stop="Fulton Street", vehicle=bus, direction="Rector Street",
departure_time=9:13pm, line=M21)

Go by the 9:13pm bus on the M21 line from Fulton Street directly to Rector Street

Adding Entrainment to Trainable NLG

- Aim: condition generation on preceding context
- Problem: data sparsity
- Solution: Limit context to just preceding user utterance
  - likely to have strongest entrainment impact
- Need for context-aware training data: we collected a new set
  - input DA
  - natural language sentence(s)
  - preceding user utterance

NEW→ *I'm headed to Rector Street*
inform(from_stop="Fulton Street", vehicle=bus, direction="Rector Street", departure_time=9:13pm, line=M21)
Go by the 9:13pm bus on the M21 line from Fulton Street directly to Rector Street

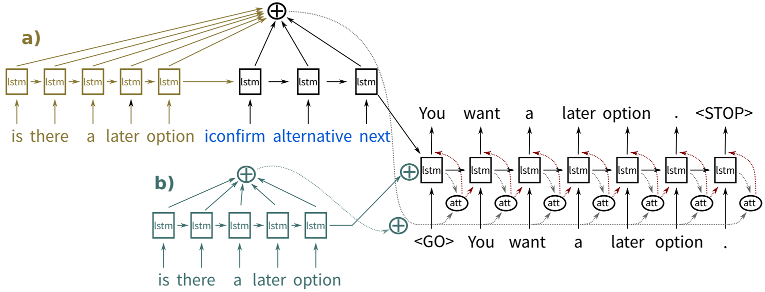
Adding Entrainment to Trainable NLG

- Aim: condition generation on preceding context
- Problem: data sparsity
- Solution: Limit context to just preceding user utterance
  - likely to have strongest entrainment impact
- Need for context-aware training data: we collected a new set
  - input DA
  - natural language sentence(s)
  - preceding user utterance

I'm headed to Rector Street
inform(from_stop="Fulton Street", vehicle=bus, direction="Rector Street",
departure_time=9:13pm, line=M21)
Heading to Rector Street from Fulton Street, take a bus line M21 at 9:13pm.

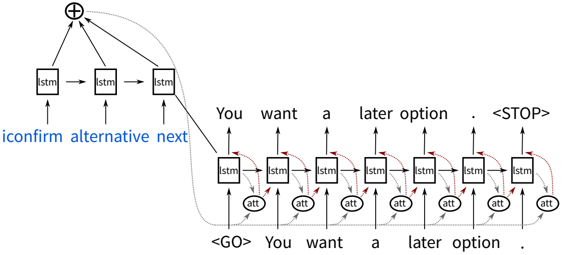
Context in our Seq2seq Generator (1)

- Two direct context-aware extensions:



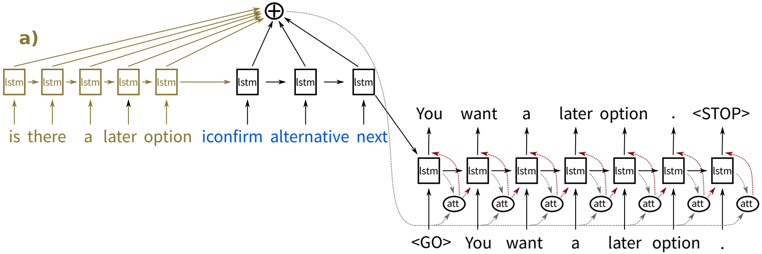
Context in our Seq2seq Generator (1)

- Two direct context-aware extensions:



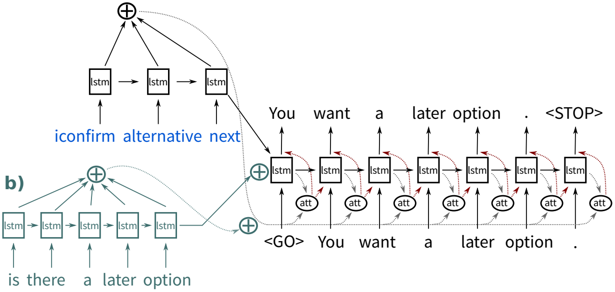
Context in our Seq2seq Generator (1)

- Two direct context-aware extensions:
  a) preceding user utterance prepended to the DA and fed into the decoder



Context in our Seq2seq Generator (1)

- Two direct context-aware extensions:
  a) preceding user utterance prepended to the DA and fed into the decoder
  b) separate context encoder, hidden states concatenated



Context in our Seq2seq Generator (2)

- One (more) reranker: *n*-gram match

Context in our Seq2seq Generator (2)

- One (more) reranker: *n*-gram match
  - promoting outputs that have a word or phrase overlap with the context utterance

is there a later time

inform_no_match(alternative=next)

-2.914	No route found later , sorry .
-3.544	The next connection is not found .
-3.690	I ' m sorry , I can not find a later ride .
-3.836	I can not find the next one sorry .
-4.003	I ' m sorry , a later connection was not found .

Context in our Seq2seq Generator (2)

- One (more) reranker: *n*-gram match
  - promoting outputs that have a word or phrase overlap with the context utterance
  - overlap measure: BLEU-2 without brevity penalty:

$$\text{logprob} \mathrel{+}= \text{weight} \cdot \sqrt{p_1 \cdot p_2}$$

is there a later time

inform_no_match(alternative=next)

-2.914	No route found later , sorry .
-3.544	The next connection is not found .
-3.690	I'm sorry , I can not find a later ride .
-3.836	I can not find the next one sorry .
-4.003	I'm sorry , a later connection was not found .

Experiments

- Dataset: public transport information
  - 5.5k paraphrases for 1.8k DA-context combinations
  - delexicalized

Experiments

- Dataset: public transport information
  - 5.5k paraphrases for 1.8k DA-context combinations
  - delexicalized

Automatic evaluation results	**BLEU**	**NIST**
Baseline (context not used)	66.41	7.037
n-gram match reranker	68.68	7.577
Prepending context	63.87	6.456
+ *n*-gram match reranker	**69.26**	7.772
Context encoder	63.08	6.818
+ *n*-gram match reranker	69.17	7.596

Experiments

- Dataset: public transport information
  - 5.5k paraphrases for 1.8k DA-context combinations
  - delexicalized

Automatic evaluation results	**BLEU**	**NIST**
Baseline (context not used)	66.41	7.037
n-gram match reranker	68.68	7.577
Prepending context	63.87	6.456
+ *n*-gram match reranker	**69.26**	7.772
Context encoder	63.08	6.818
+ *n*-gram match reranker	69.17	7.596

- Human pairwise preference ranking (crowdsourced)
  - baseline × prepending context + *n*-gram match reranker

Experiments

- Dataset: public transport information
  - 5.5k paraphrases for 1.8k DA-context combinations
  - delexicalized

Automatic evaluation results	**BLEU**	**NIST**
Baseline (context not used)	66.41	7.037
n-gram match reranker	68.68	7.577
Prepending context	63.87	6.456
+ *n*-gram match reranker	**69.26**	7.772
Context encoder	63.08	6.818
+ *n*-gram match reranker	69.17	7.596

- Human pairwise preference ranking (crowdsourced)
  - baseline × prepending context + *n*-gram match reranker
  - context-aware preferred in 52.5% cases (significant)

Output Examples

Context	is there a later option
Input DA	iconfirm(alternative=next)
Baseline	Next connection.
n-gram match reranker	You want a later connection.
Prepending context + *n*-gram match reranker	You want a later connection.
Context encoder + *n*-gram match reranker	You want a later option.

Output Examples

Context	is there a later option
Input DA	iconfirm(alternative=next)
Baseline	Next connection.
n-gram match reranker	You want a later connection.
Prepending context + *n*-gram match reranker	You want a later connection.
Context encoder + *n*-gram match reranker	You want a later option.

Context	i need to find a bus connection
Input DA	inform_no_match(vehicle=bus)
Baseline	No bus found, sorry.
n-gram match reranker	I did not find a bus route.
Prepending context + *n*-gram match reranker	I'm sorry, I cannot find a bus connection.
Context encoder + *n*-gram match reranker	I'm sorry, I cannot find a bus connection.

Output Examples

Context	i rather take the bus
Input DA	inform(vehicle=bus, departure_time=8:01am, direction=Cathedral Parkway, from_stop=Bowling Green, line=M15)
Baseline	At 8:01am by bus line M15 from Bowling Green to Cathedral Parkway.
n-gram match reranker	At 8:01am by bus line M15 from Bowling Green to Cathedral Parkway.
Prepending context + *n*-gram match reranker	You can take the M15 bus from Bowling Green to Cathedral Parkway at 8:01am.
Context encoder + *n*-gram match reranker	At 8:01am by bus line M15 from Bowling Green to Cathedral Parkway.

Conclusion

- 1st fully trainable NLG system for dialogue systems capable of adapting to previous user utterances

Conclusion

- 1st fully trainable NLG system for dialogue systems capable of adapting to previous user utterances
- significant BLEU improvement over baseline

Conclusion

- 1st fully trainable NLG system for dialogue systems capable of adapting to previous user utterances
- significant BLEU improvement over baseline
- confirmed in human evaluation

Future Plans

- Longer context

Conclusion

- 1st fully trainable NLG system for dialogue systems capable of adapting to previous user utterances
- significant BLEU improvement over baseline
- confirmed in human evaluation

Future Plans

- Longer context
- Fuzzy *n*-gram matching

Conclusion

- 1st fully trainable NLG system for dialogue systems capable of adapting to previous user utterances
- significant BLEU improvement over baseline
- confirmed in human evaluation

Future Plans

- Longer context
- Fuzzy *n*-gram matching
- Avoiding delexicalization

Conclusion

- 1st fully trainable NLG system for dialogue systems capable of adapting to previous user utterances
- significant BLEU improvement over baseline
- confirmed in human evaluation

Future Plans

- Longer context
- Fuzzy *n*-gram matching
- Avoiding delexicalization
- Integrate into an end-to-end SDS

Thank you for your attention

Download it!

- Code: `bit.ly/tgen_nlg`
- Dataset: `bit.ly/nlgdata`

Contact me

Ondřej Dušek
o.dusek@hw.ac.uk