Conditional Text Generation

Dr. Nurul Lubis

Dialog Systems and Machine Learning Group

25.06.2021

Generative Networks

- Today's generative networks are trained on huge amounts of data
  - Generalizability to unseen samples
- Example: chit-chat systems
  - Trained on data from reddit, twitter, ...
- Once deployed, we have no control over the generation

Role	response
User	Does money buy happiness?
Bot	Depends how much money you spend on it .
User	What is the best way to buy happiness ?
Bot	You just have to be a millionaire by your early 20s, then you can be happy .
User	This is so difficult !
Bot	You have no idea how hard it is to be a millionaire and happy . There is a reason the rich have a lot of money

DialoGPT (Zhang et al., 2019)

Conditional Text Generation

Heinrich Heine
Universität Düsseldorf

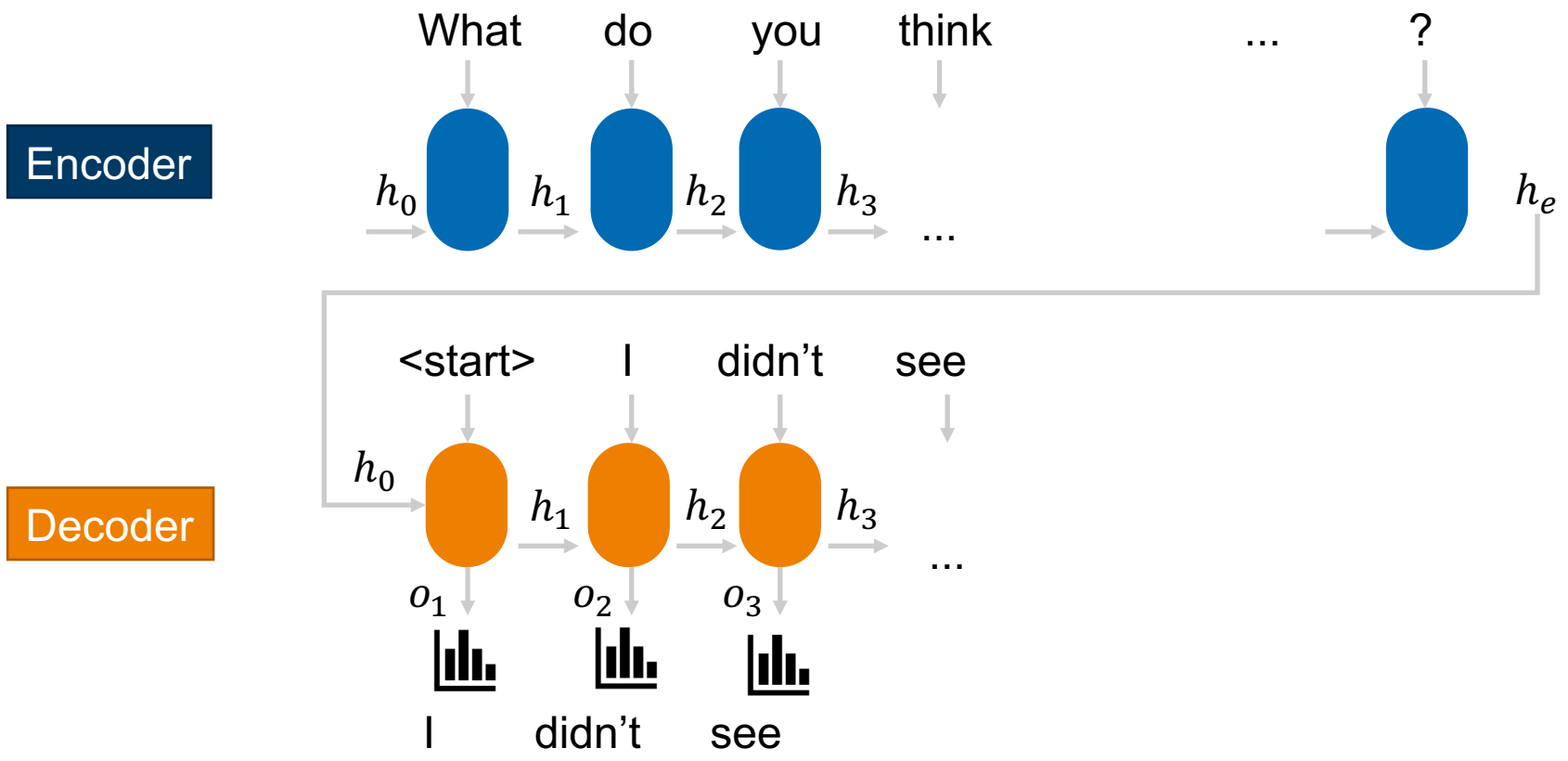
- Generation with one or more controllable attributes
  - Generation is *conditioned* on the control attribute
  - Adding adjustable “knobs” in the generation process

Conditional Text Generation

- Generation with one or more controllable attributes
  - Generation is *conditioned* on the control attribute
  - Adding adjustable “knobs” in the generation process
- Gaining popularity, many possible applications
  - Story generation
    - Ending (Peng et al., 2018) and plot (Yao et al., 2019)
  - E-mail
    - Politeness (Madaan et al., 2020)
  - Dialogue
    - Chitchat: Persona (Zhang et al., 2018; Dinan et al., 2016)
    - Task-oriented: dialogue action

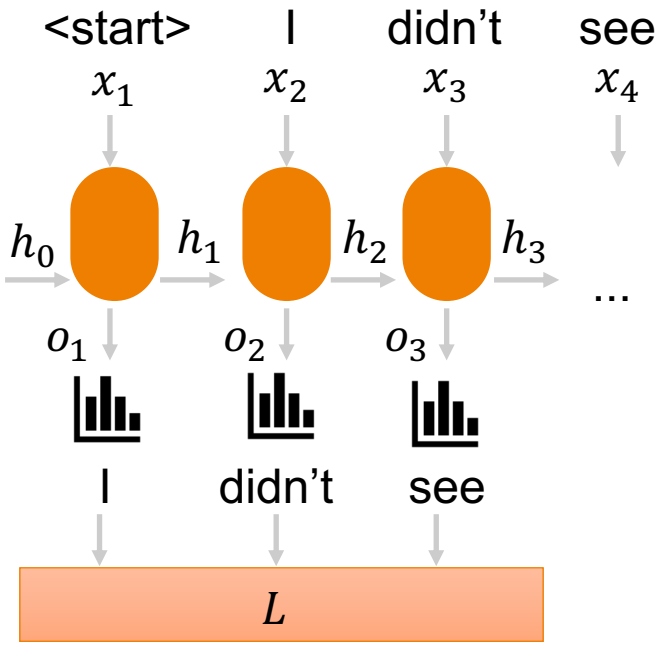
Text Generation

- Mainly framed as a language generation task
- Recurrent NN:



Conditional Text Generation

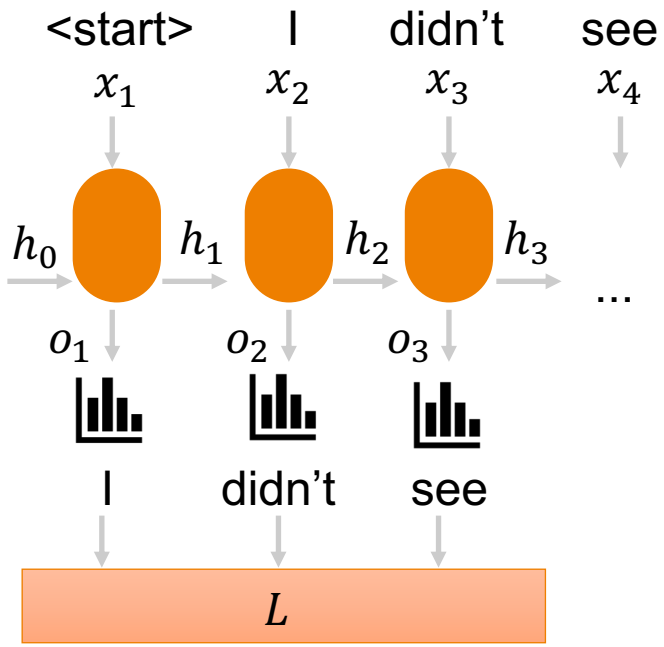
- Where can we add control attributes in the decoding process?



- 5 modules in the modification space (Prabhumoye et al., 2020)
  - External input h_0
    - Start of decoding or generation
  - Sequential input x
    - Control at every time step
  - Generator operation
    - Parameterizing the control attribute
  - Output o
    - Projection to vocabulary
  - Loss function L
    - Comparison to target

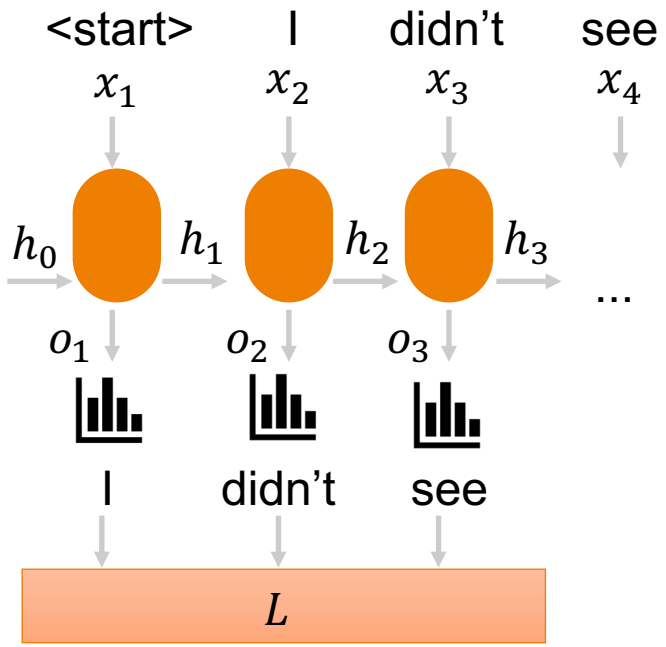
External input, initialization of decoder state

- Arithmetic or linear transformation
  - When the control attribute comes from another source, it can be combined into h_0
    - Linear layer (Hoang et al., 2016)
    - Arithmetic operation (Chandu et al., 2019)
    - Concatenation (Fu et al., 2018, Zhou et al., 2018; Dinan et al., 2018)
- Stochastic changes
  - Variational model such as VAE (Kingma and Welling, 2014)
    - Sample a latent variable to be used to initialize the decoding process (Bowman at al., 2016)
- External feedback
  - Put h_e through a discriminator to detect the control signal
  - Optimize the representation



Sequential input

- Giving control signal at every time step
- Similar to before, the control attribute can be combined with each step of the sequential input with arithmetic or linear transformation
- Numerous works based on RNN in different domains do not show impressive results
  - Word definition: word to be defined is given as input at every time step concatenated with previously generated token (Noraset et al., 2017)
  - Article generation: hidden state of encoder is concatenated at every time step (Phrabumoye et al., 2019)



Generator operation, mapping input to output

Heinrich Heine
Universität Düsseldorf

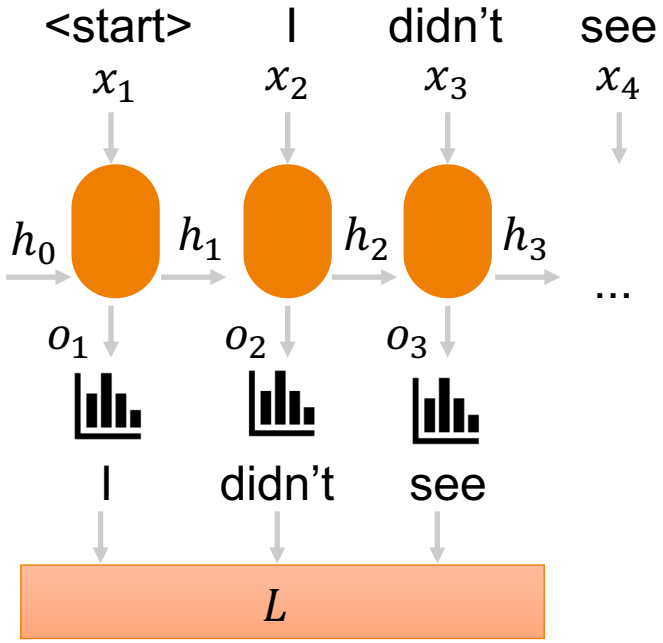
- Recurrent NNs (Rumelhart et al., 1986)
  - RNN cells contain different gates and operation to better model context
    - LSTM (Hochreiter and Schmidhuber, 1997), GRU (Cho et al., 2014)
  - Similarly, operations can be added to compute the control attribute
  - Factored LSTM (Gan et al., 2017) for image captioning
    - LSTM parameters are factored into three components, each responsible for style, image, and its associated tag.
  - Semantic conditioning (Wen et al., 2015)
    - add a cell in addition to LSTM cell which takes dialogue action to perform sentence planning.
- Transformers (Vaswani et al., 2017)
  - Rely on attention mechanisms to draw global dependencies between input and output
  - Hierarchical disentangled attention for graph-based conditioning (Chen et al., 2019)
- Pre-trained models
  - Fine-tuning pre-trained models for downstream tasks has shown good results
  - Plug and Play LM (Dathathri et al., 2019) combines pre-trained LM with classifiers to guide the generation process.
  - Changes made should not interfere with pre-trained weights otherwise retraining will be necessary

Output

- Attention
  - Guiding the generation process by attending to the source sequence (Bahdanau et al., 2015)
    - Find a context vector at *each time step* which captures the information needed from the source
  - Attention for controlled generation
    - Add style control attribute to input sequence (Sudhakar et al., 2019)
    - Use attention to attend over external document as additional source (Dinan et al., 2018)
    - Use the representation of agent persona to compute attention weights (Zhang et al., 2018)
      - Attention weights is recomputed according to the control attribute
- External feedback
  - The output space can be modified with adversarial loss
    - An adversary tries to distinguish sentence whose style has been transferred or sentence from the true target distribution (Shen et al., 2017)
  - Use reward to optimize the output
    - Style reward, semantic reward, and fluency reward (Gong et al., 2019)
- Arithmetic or linear transform
  - Combine the output with control attribute through a linear layer, addition, or concatenation (Hoang et al., 2016)
    - The transformed output is then used to predict the target token

Loss function

- In addition to typical loss for matching to target, we can add loss to optimize for the control attribute
- KL loss
  - Measure of divergence between two distributions
  - Variational models.
    - If prior for the (latent) control variable is known, we can use KL loss to minimize the distance between the learned posterior distribution and the prior (Lubis et al., 2020)
- Classifier loss
  - Ensure that the generated sequence comply with the control attribute
    - Which style is this sequence? (Hu et al., 2017; Phrabumoye et al., 2018; Sudhakar et al., 2019)
    - Does this sequence belong to style A? (Chandu et al., 2019)
    - Does this sequence have the same style as sequence sampled from a particular style? (Yang et al., 2018)
- Custom task specific loss



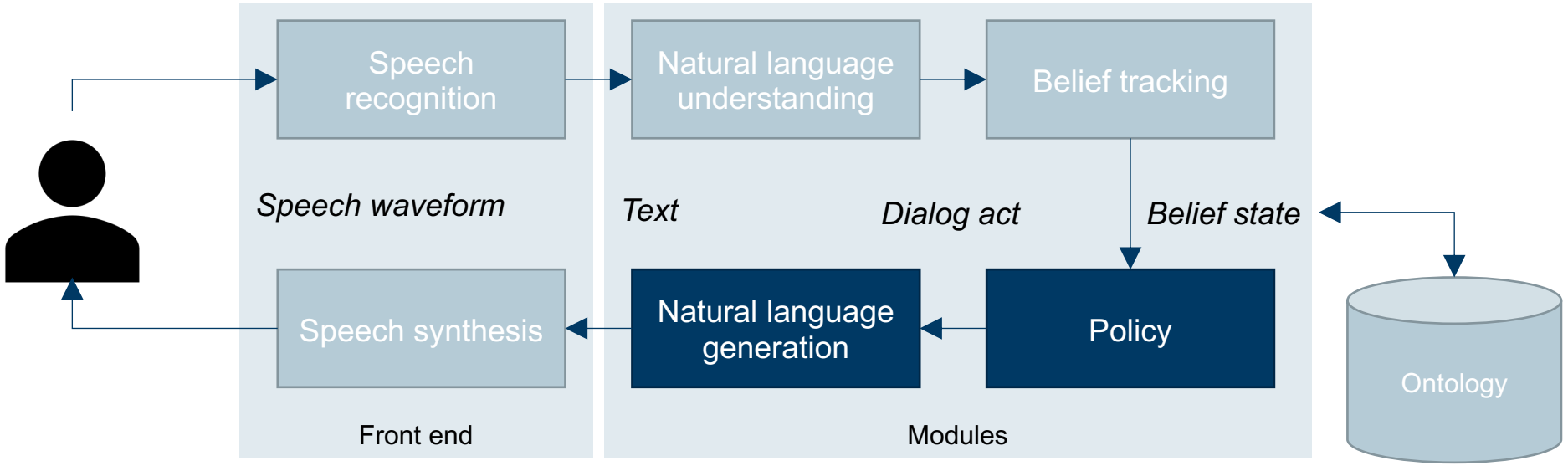
Heinrich Heine
Universität Düsseldorf

Controlled Generation in Dialogue Systems

Semantic action as control

Heinrich Heine
Universität Düsseldorf

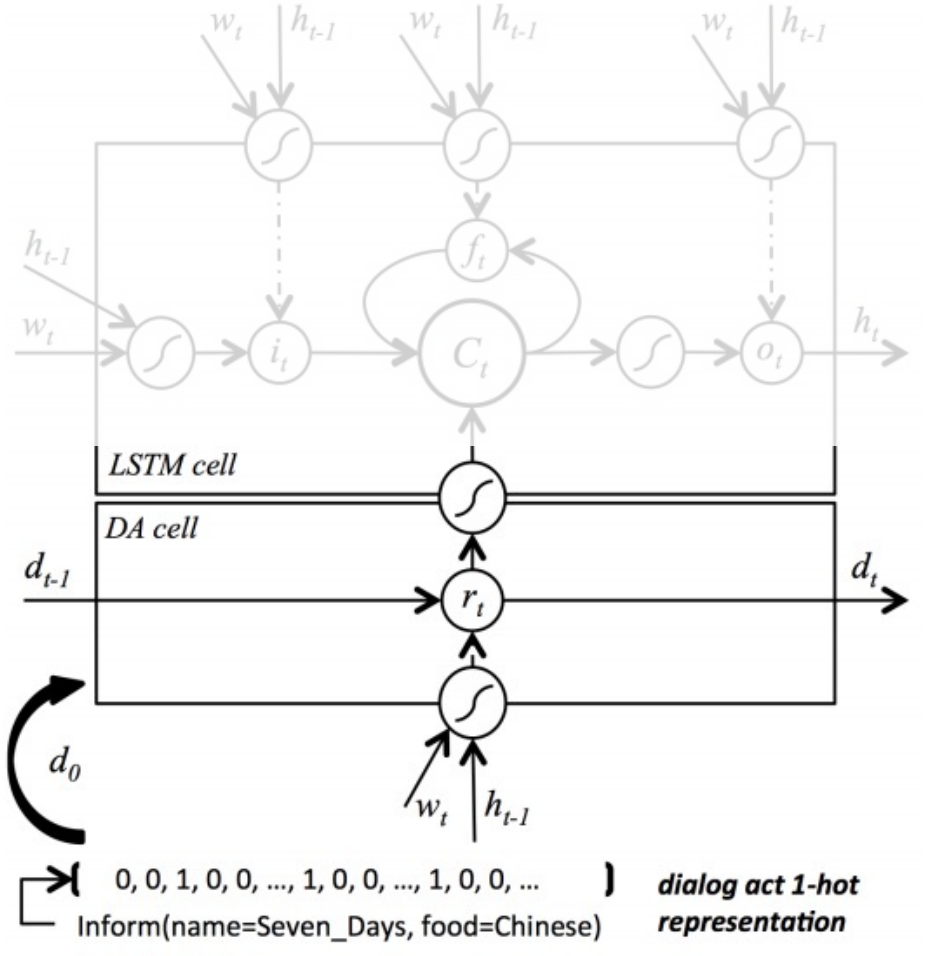
Turn 1:	I'm looking for an Italian restaurant	inform(food=Italian)	food=Italian
Turn 2:	Eastern part of town	inform(area=East)	food=Italian, area=East



Which area do you have in mind?	request(area)
Which price range would you like?	request(price)

We would like to generate natural language response conditioned on the dialogue action

Semantically-conditioned LSTM (Wen et al., 2015)



- LSTM cell
  - input, forget, and output gates
  - Input is current token and previous hidden state
- Dialogue action (DA) cell below LSTM cell
  - Input is dialogue action, modifies the dialogue act
    - at each time step t the DA cell *reading gate* decides what information should be retained for future time steps and discards the others

$$\mathbf{r}_t = \sigma(\mathbf{W}_{wr}\mathbf{w}_t + \alpha\mathbf{W}_{hr}\mathbf{h}_{t-1}) \quad (7)$$

$$\mathbf{d}_t = \mathbf{r}_t \odot \mathbf{d}_{t-1} \quad (8)$$

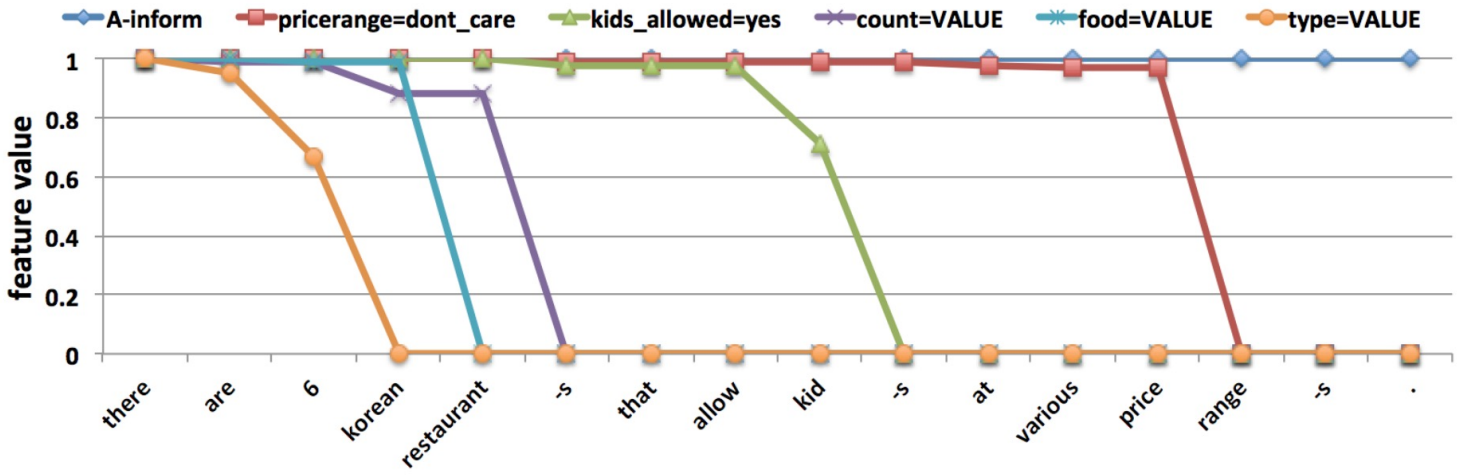
  - Performs sentence planning
    - Manipulates the LSTM value based on dialogue act features

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{c}}_t + tanh(\mathbf{W}_{dc}\mathbf{d}_t) \quad (9)$$

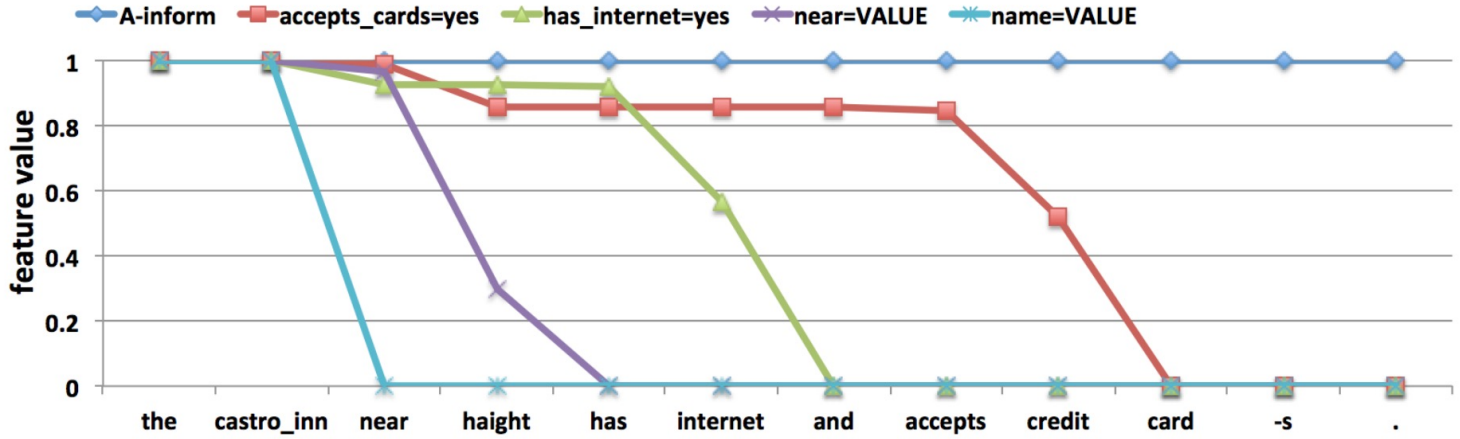
- Skip connection, backwards reranking

Semantically-conditioned LSTM (Wen et al., 2015)

- Tested on restaurant and hotel domains
  - 8 shared system dialogue act
  - 12 slots per domain with some overlap
  - 248 distinct DA for restaurant domain, 164 for hotel
- Objective and subjective evaluations compared to RNN
  - More natural and informative
  - More preferred by humans



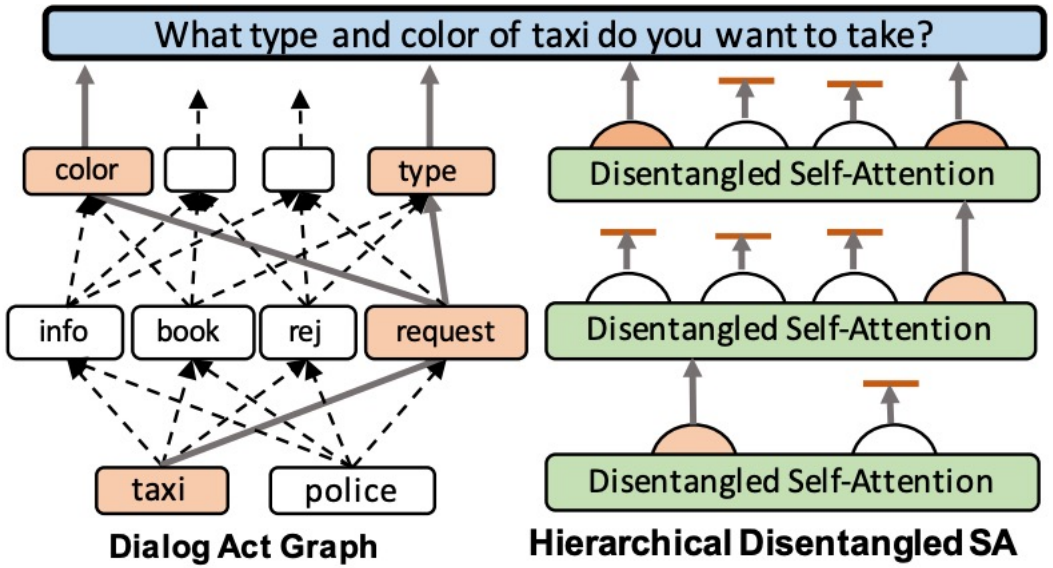
(a) An example realisation from SF restaurant domain



(b) An example realisation from SF hotel domain

Graph-based generation (Chen et al., 2019)

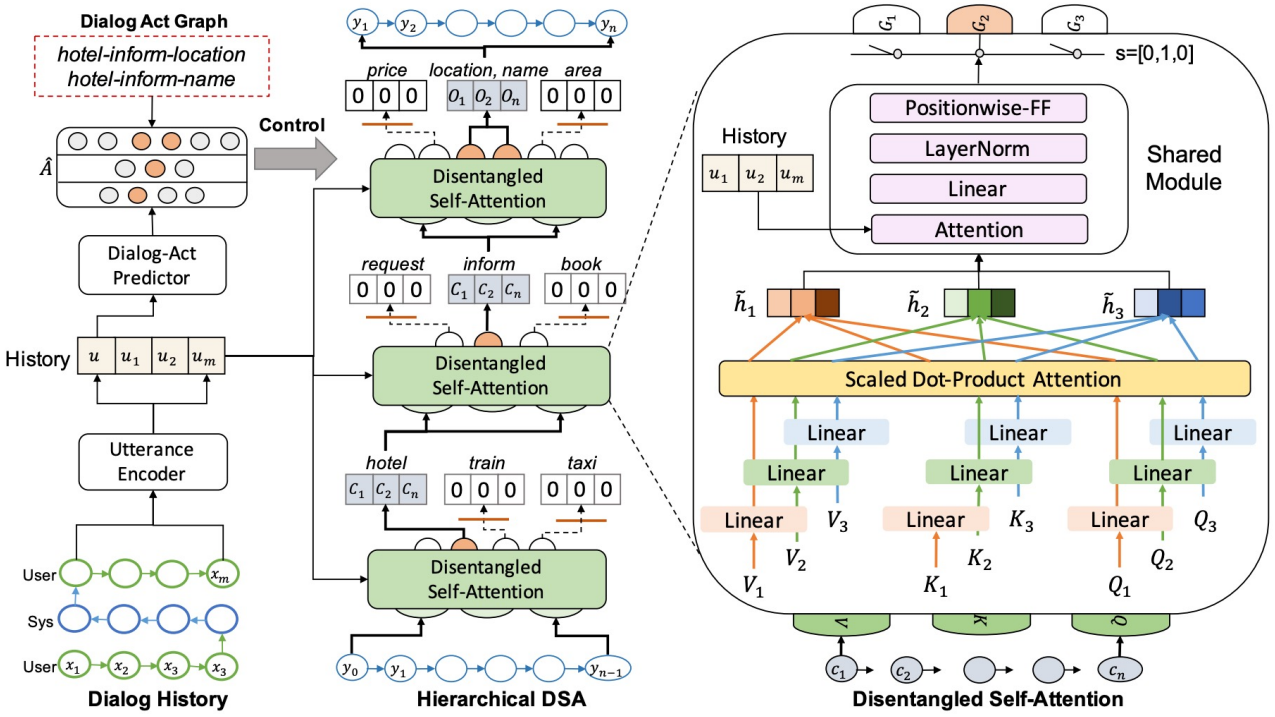
- Multi-domain large scale ontology requires scalability
  - Graph-based representation exploits commonalities between domains, e.g. intents and slots
- Hierarchical disentangled self-attention
  - Different attention heads focus on different nodes in the graph
    - Hierarchical: Layers in the graph
    - Domain -> action -> slot
  - Allows efficient combinations of nodes



Disentangled self-attention (DSA)

Heinrich Heine
Universität Düsseldorf

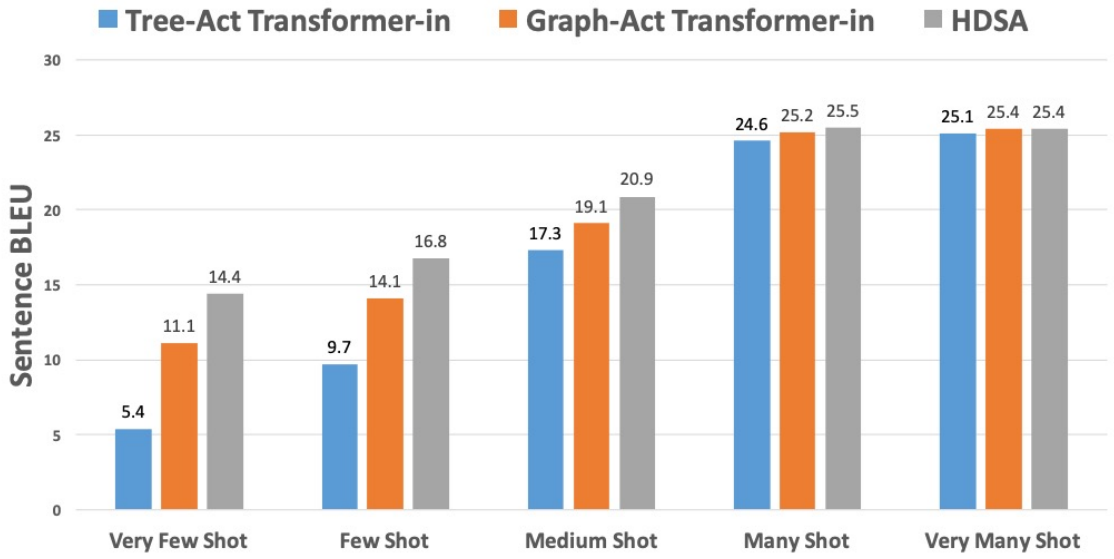
- Transformers concatenate vectors from multiple heads into the final vector
- While the method to compute attention in each head is identical to transformer, DSA employs binary vector which acts as a switch
  - Only vectors from “active” heads are considered
  - The active vectors are summed together
- Multiple DSA layers are stacked to better handle complexity



Graph-based generation (Chen et al., 2019)

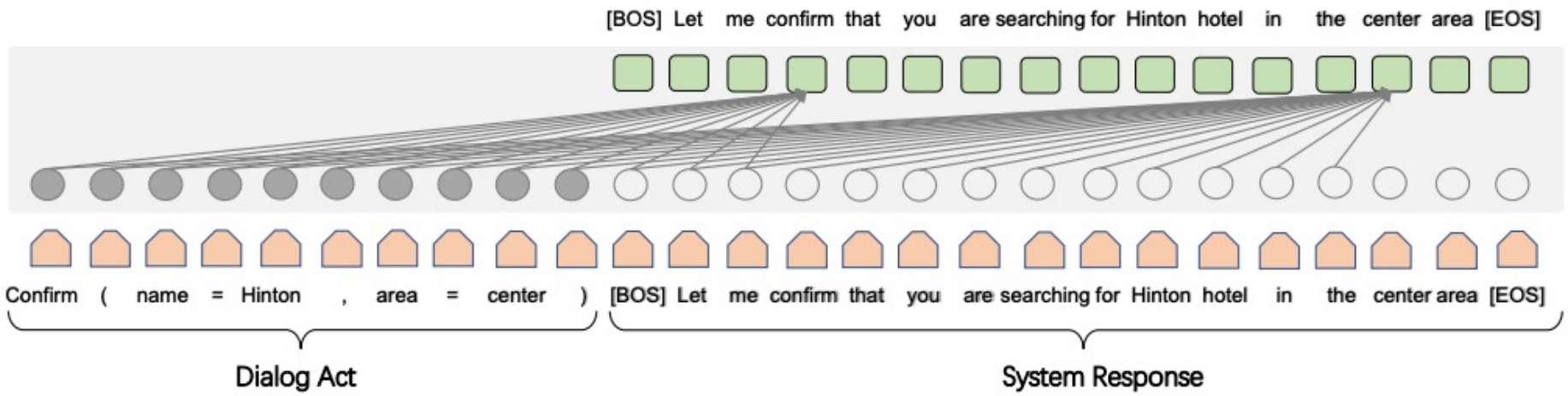
Heinrich Heine
Universität Düsseldorf

- MultiWOZ data (Budzianowski et al., 2018)
- Training
  - Dialogue act is used to activate corresponding heads, creating a path inside the graph.
  - The generated response is matched to the target to compute loss for optimization
- Few-shot experiment
  - Graph-act representation helps generalization in few-shot cases
  - HDSA further exploit the hierarchical structure of the dialogue action space



Semantically-conditioned GPT (Peng et al., 2020)

- Leveraging the fluency of large pre-trained models
  - Pre-trained GPT2 Medium, 354M parameters
  - Fine-tune with dialogue NLG task
- Training: Dialogue act followed by system response
- Inference: Dialogue act followed by [BOS] token



Few-shot study on SC-GPT (Peng et al., 2020)

- FewShotWOZ:
  - A modified RNNLM (Wen et al., 2015) and MultiWOZ (Budzianowski et al., 2018) corpus with smaller training set, more domains, and smaller overlap between train and test sets.

Model	Restaurant		Laptop		Hotel		TV		Attraction		Train		Taxi	
	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓
SC-LSTM	15.90	48.02	21.98	80.48	31.30	31.54	22.39	64.62	7.76	367.12	6.08	189.88	11.61	61.45
GPT-2	29.48	13.47	27.43	11.26	35.75	11.54	28.47	9.44	16.11	21.10	13.72	19.26	16.27	9.52
SC-GPT	**38.08**	**3.89**	**32.73**	**3.39**	**38.25**	**2.75**	**32.95**	**3.38**	**20.69**	**12.72**	**17.21**	**7.74**	**19.70**	**3.57**

Table 3: Performance of different methods on FEWSHOTWOZ

Model	Entity F1	BLEU
SC-LSTM (Wen et al., 2015b)	80.42	21.6
HDSA (Chen et al., 2019)	87.30	26.48
GPT-2	87.70	30.71
SC-GPT	**88.37**	**30.76**

Table 5: Performance on MultiWOZ

Model	Seen		Unseen	
	BLEU ↑	ERR ↓	BLEU ↑	ERR ↓
SC-LSTM	23.05	40.82	12.83	51.98
GPT-2	30.43	3.26	27.92	17.36
SC-GPT	**40.28**	**1.09**	**36.69**	**4.96**

Table 9: Performance of different methods on seen DAs and unseen DAs in restaurant domain.

Plug and Play Conversational Models (Madotto et al., 2020)

Heinrich Heine Universität Düsseldorf

- Chit-chat system
  - Trained on large amounts of data
  - Benefits from large pre-trained models
    - DialoGPT (Zhang et al., 2019)
- Plug and Play Conversational Models
  - Residual Adapters (Houlsby et al., 2019; Bapna and Firat, 2019) within DialoGPT
    - Added on top of each transformer layer
    - Steers the output distribution without modifiying the pre-trained weights

$$f_{\theta_i}(x) = \texttt{ReLU}\big(\texttt{LN}(x) \cdot W_i^E\big) \cdot W_i^D,$$

$$\texttt{Adapter}(o_{:t}^i) = f_{\theta_i}(o_{:t}^i) + o_{:t}^i,$$

    - One adapter per control attribute

- Fine-tuning
  - Given a set of dialogues with certain attribute a, optimize the parameter of the residual adapters to minimize the NLL over the dataset
  - The parameters of the large pre-trained model is untouched
  - The data used for fine-tuning need not be dialogue data, and can be artificially generated

					Score by Attribute				
	↓ **Ppl.**	↑ **Dist 1/2/3**	**Discrim.**	**Score**	**Posi.**	**Nega.**	**Busin.**	**Sci/Tech**	**Sport**
DG	**39.60**	0.22/0.64/0.77	46.48	32.91	65.67	19.40	17.41	91.04	27.86
WD	53.03	0.25/0.74/**0.84**	50.18	34.54	58.21	28.86	19.40	91.04	36.82
PP	45.86	0.24/0.67/0.79	73.28	49.54	75.12	51.74	47.26	93.03	59.20
AD	41.57	0.17/0.58/0.77	**96.52**	**70.01**	**93.03**	**73.13**	**68.66**	**99.00**	**83.08**

Table 3: Automatic evaluation results. In all the metrics higher is better except for Perplexity (Ppl.), and *Discrim.* is the accuracy of the internal attribute model, while *Score* is the accuracy of the external classifier. All the results, are averaged among the six attribute models.

Conclusion

Heinrich Heine
Universität Düsseldorf

- Conditional generation is gaining attention
  - Many ways to Rome
    - Conditioning can be done at various places in the generation process
  - Wide range of applications
    - Dialogue systems are natural benefactors
- Generalizability
  - How we represent the control attributes
  - Parameter sharing
- Benefiting from large pre-trained models
  - How to minimize re-training of pre-trained parameters
  - There may be some tradeoff with fluency
    - How to maintain fluency while perturbing the generation

Heinrich Heine
Universität Düsseldorf

Thank you!