Neural Networks for Natural Language Understanding

Gabriel Roccabruna

Signals and Interactive Systems Lab

Department of Information Engineering and Computer Science

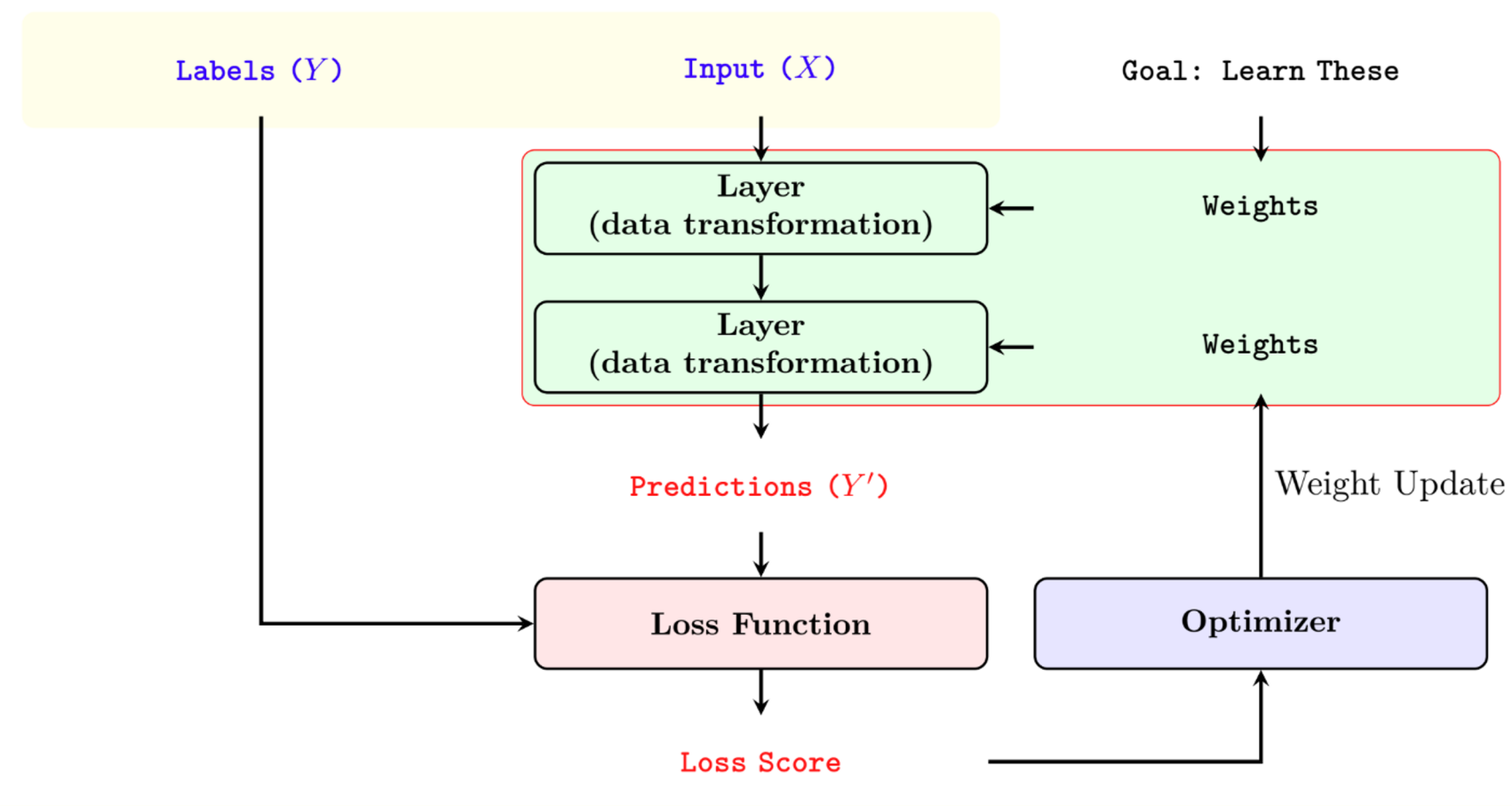
University of Trento

gabriel.roccabruna@unitn.it

Outline

- Neural Network introduction
- Recurrent Neural Network
- Transformers
- Natural Language Understanding
  - Models and Datasets

How do Neural Networks work?



NN Training

- Loss Function: cross-entropy loss
- Optimization: Gradient Descent
  - need to know the gradient of the loss function
  - Backpropagation is used to compute the partial derivates to calculate the gradient of each weight

Loss Function

- The loss function computes the distance between the current (prediction) and expected (ground truth) output

- Popular NN Loss Functions:
  - Binary Cross-Entropy
    - for binary classification
  - Multi-class Cross Entropy
    - For multi-class problems

Regularization

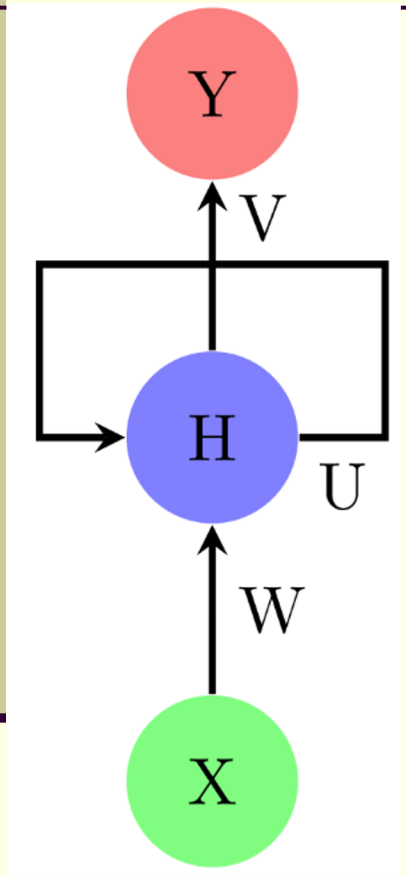
- Regularization is a technique for preventing overfitting
  - Minimizes the validation loss
  - Adds penalty to the model with high variance
- In Logistic Regression, we use L1 and L2 regularization
- Dropout is a regularization technique applied to NN that ignores randomly selected neurons during training

Recurrent Neural Networks

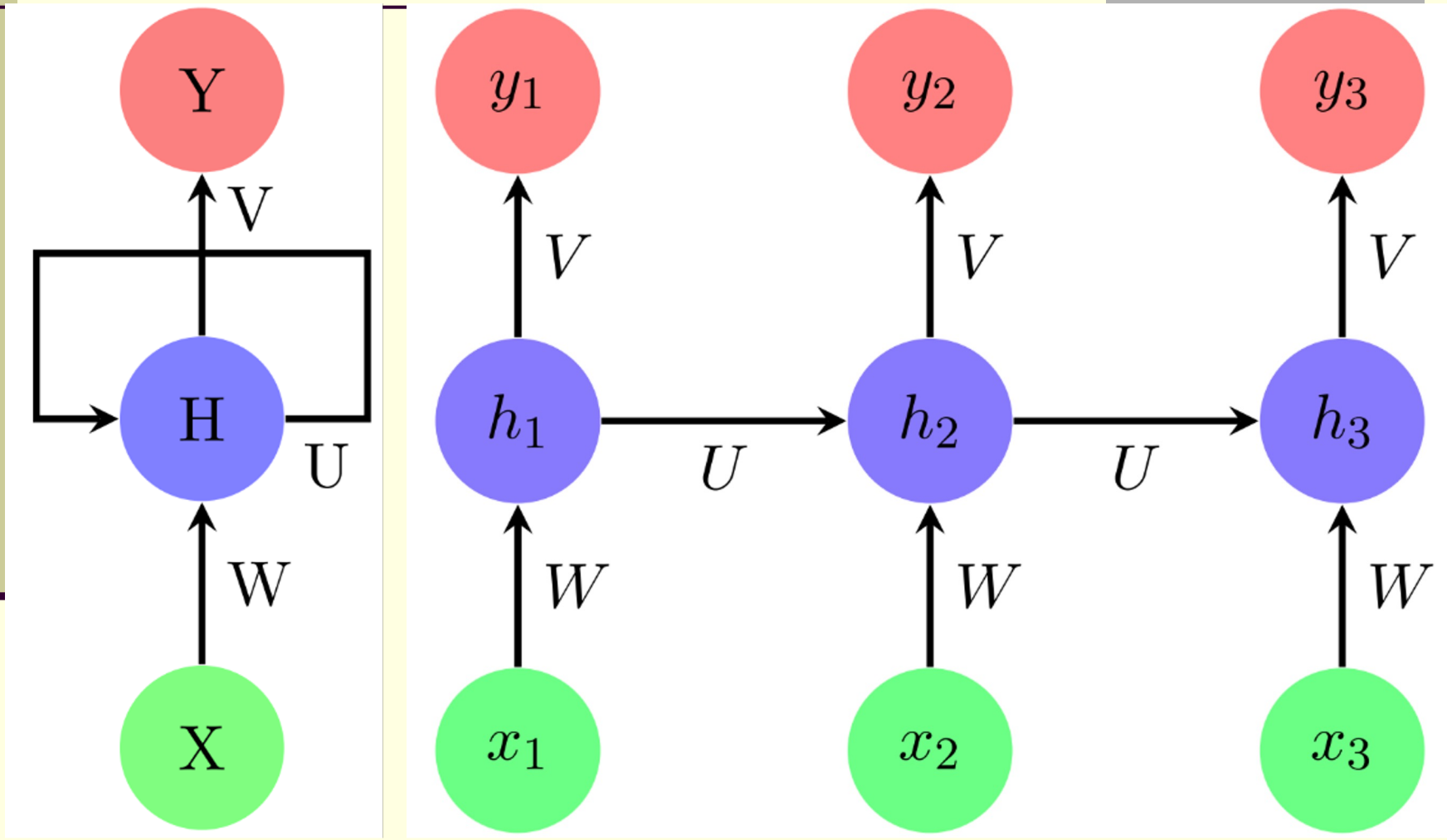
Recurrent Neural Network

- A recurrent neural network (RNN) is any network that contains a cycle within its network connections. That is, any network where the value of a unit is directly, or indirectly, dependent on its own earlier outputs as an input.

Recurrent Neural Network



Recurrent Neural Network



RNN vs NN

- Additional set of weights from computing the hidden state (*U*)
- Weights are shared across time (W, V, U):
  - $h_t = g(Wx_t + U\, h_{t-1} + b)$
  - $Y_t = f(V\, h_t) = \text{softmax}(V\, h_t)$

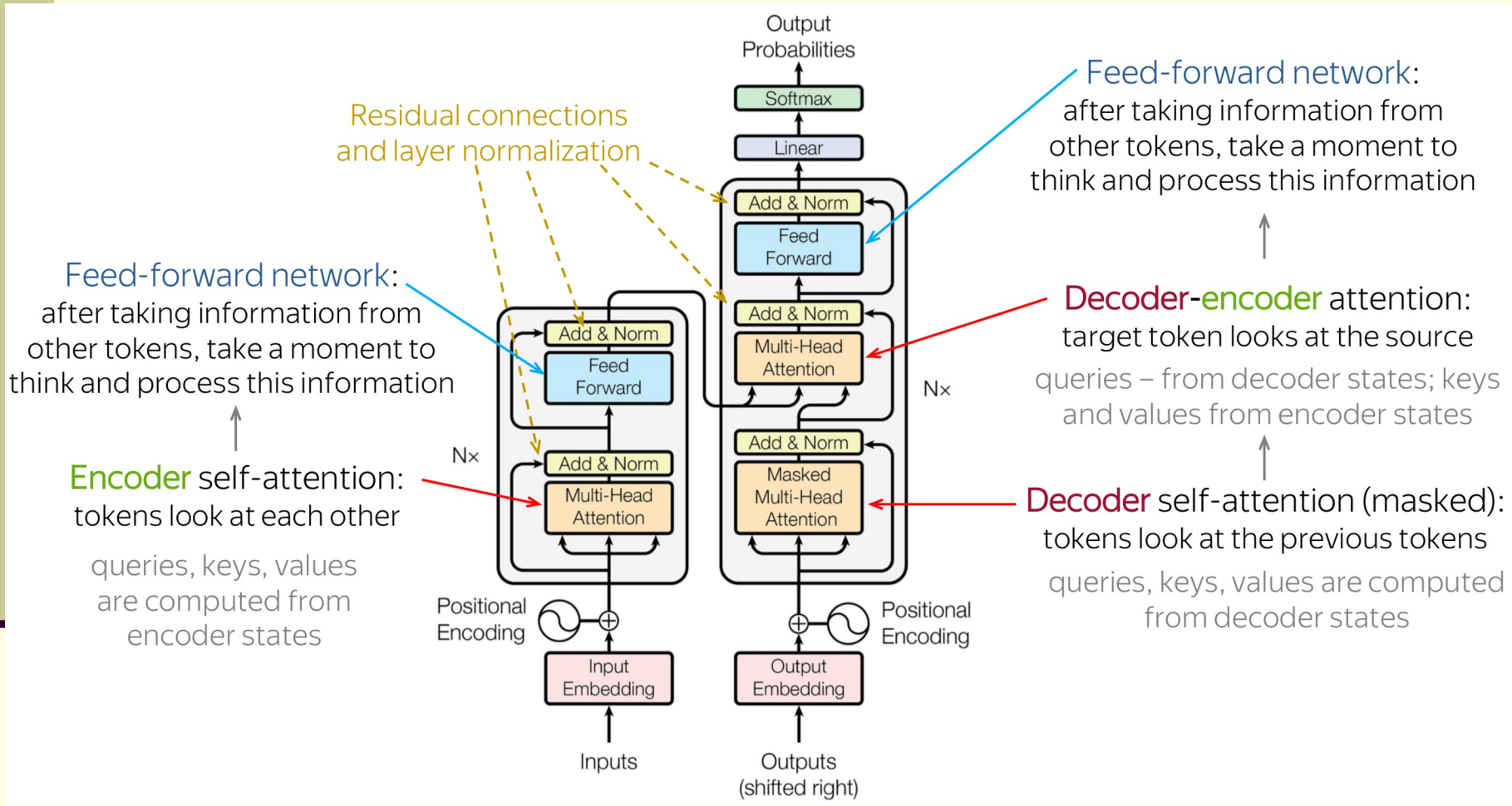
The problem of Long-Term Dependencies

- RNN can only exploit recent information to solve the present task
- As the distance increases RNNs are unable to learn connected information
  - LSTM (Long Short Term Memory)
  - GRU (Gated Recurrent Units)

Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780. (LSTM)
Cho, Kyunghyun, et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation." (GRU)

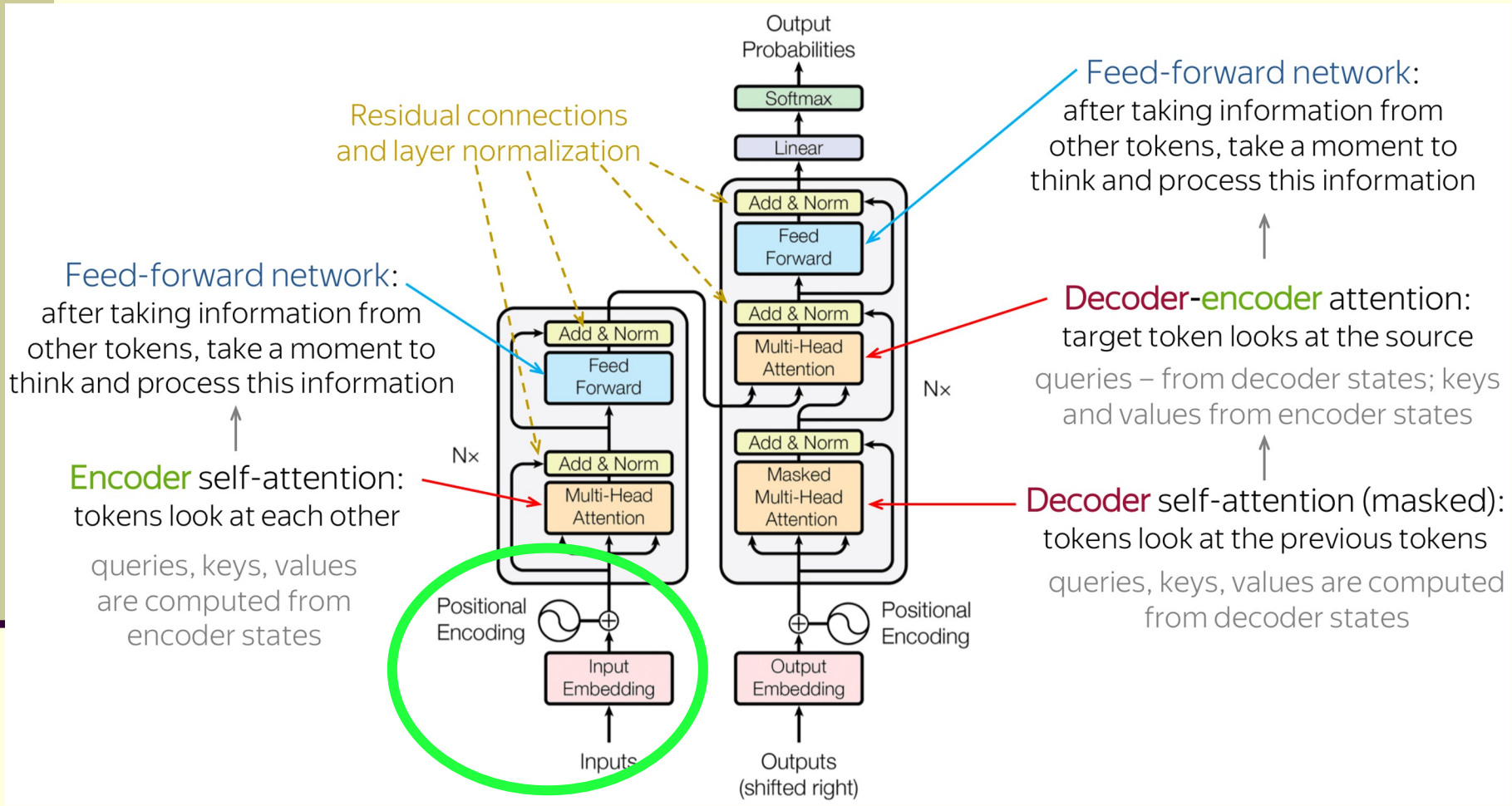
Transformers

Transformer Architecture



Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

Transformer Architecture



Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

Encoder - Input Embeddings

- **Tokenization**: byte-pair encoding (sub-token level)
- **Token embeddings**: learned during training
- **Positional encodings**: additive embeddings used to give an order to the elements of a sequence

Tokenization: Byte-Pair Encoding

Initialisation

Tokenize the corpus with white space and the tokens at the character level

The initial vocabulary is the set of all the characters

- Decide a vocabulary size S

Repeat until the vocab is equal to the target size S

- Find the most frequent pair of symbols X and Y in the corpus
- Create a new symbol XY and add it to the vocab
- Substitute all the instances of the pair of symbols X, Y with the new symbol XY

Byte-Pair Encoding - Example

a sailor went to sea, sea, sea
to see what he could see, see, see.
but all that he could see, see, see
was the bottom of the deep blue sea, sea, sea.

Vocabulary: {a, b, c, …, z}

Byte-Pair Encoding - Example

a sailor went to sea, sea, sea
to see what he could see, see, see.
but all that he could see, see, see
was the bottom of the deep blue sea, sea, sea.

Vocabulary: {a, b, c, …, z}

The most frequent pair of symbols: s,e
X = s and Y = e

Byte-Pair Encoding - Example

a sailor went to sea, sea, sea
to see what he could see, see, see.
but all that he could see, see, see
was the bottom of the deep blue sea, sea, sea.

Vocabulary: {a, b, c, …, z, SE}

The most frequent pair of symbols: s,e
X = s and Y = e

- Create a **new symbol** SE and add it to the vocabulary

Byte-Pair Encoding - Example

a sailor went to SEa, SEa, SEa
to SEe what he could SEe, SEe, SEe.
but all that he could SEe, SEe, SEe
was the bottom of the deep blue SEa, SEa, SEa.

Symbol Vocabulary: {a, b, c, …, z, SE}

The most frequent pair of symbols: s,e

- Create a **new symbol** SE and add it to the vocabulary
- **Substitute** all instances of the pair s,e with the new symbol SE

Byte-Pair Encoding - Example

Vocabulary: {a, b, c, …, z, SE, SEE, SEA, HE, TO}

He will be back in town and on set for the next season

Byte-Pair Encoding - Example

Vocabulary: {a, b, c, …, z, SE, SEE, SEA, HE, TO}

He will be back in town and on set for the next season

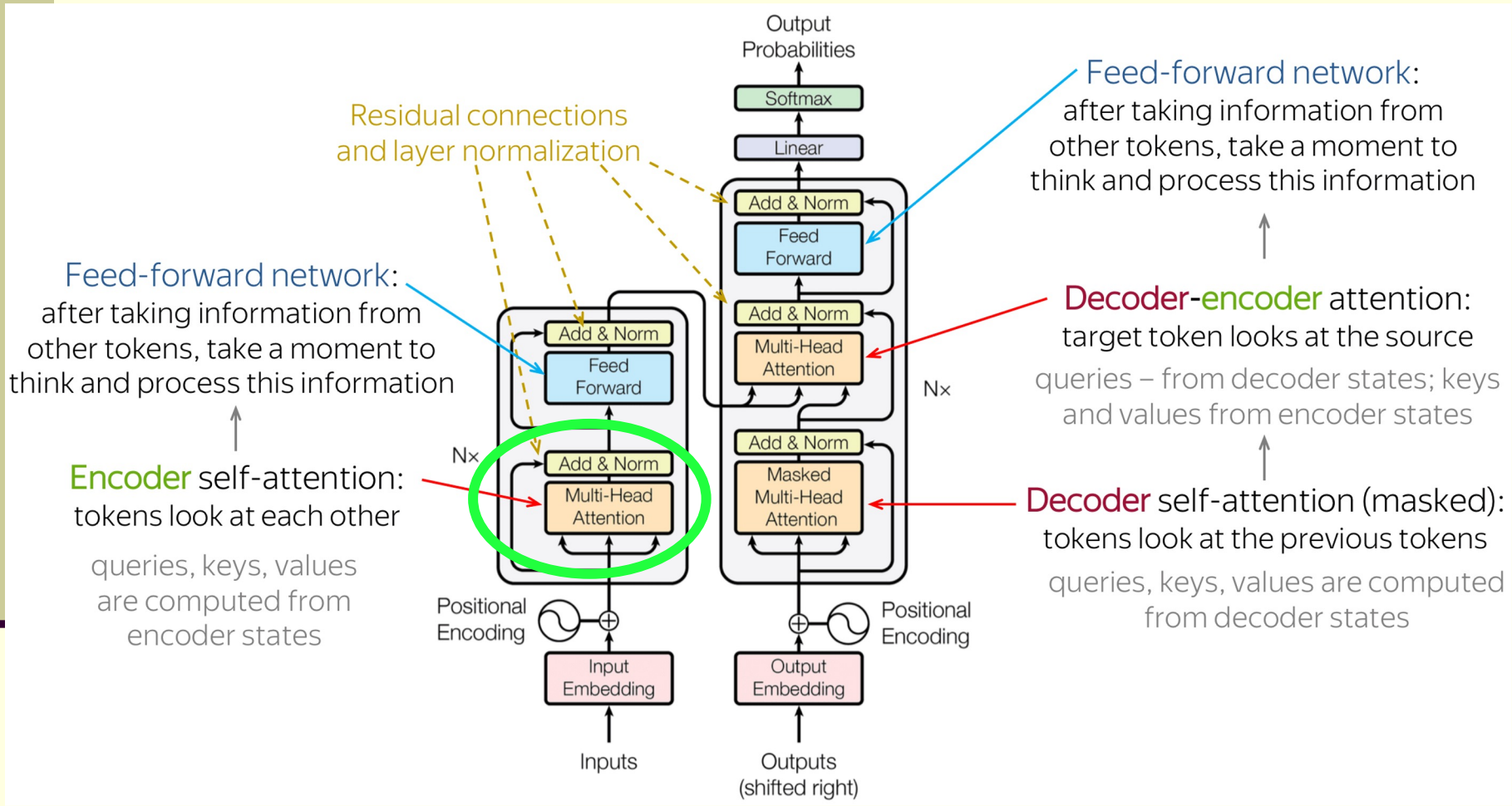
HE will be back in TOwn and on SEt for tHE next SEAson

HE w i l l b e b a c k i n TO w n a n d o n SE t f o r t HE n e x t SEA s o n

Season = SEA ##s ##o ##n

Town = TO ##w ##n

Transformer Architecture



Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

Self-Attention Mechanism

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^\top}{\sqrt{d_k}})V$$

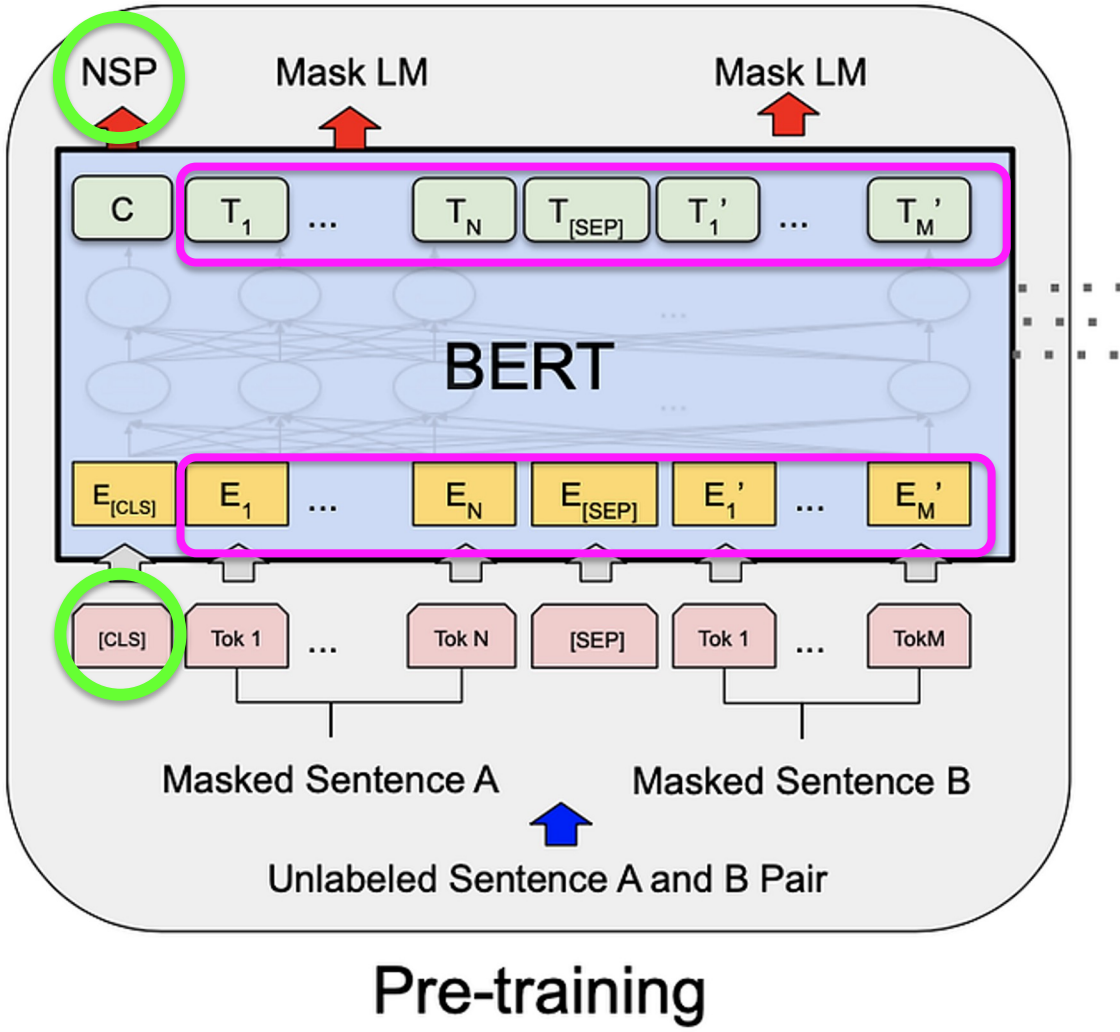
- Q: Query ($W_Q x$)
- K: Key ($W_K x$)
- V: Value ($W_V x$)
- d_k is the dimension of Q and K
- X: it's the input sequence

Transformer Base model in NLP

- Bidirectional Encoder Representation from Transformer (**BERT**)
  - Masked-language Model
- Generative pre-trained Transformer (**GPT**)
  - Language model based on the Decoder of the transformer
- Text-To-Text Transfer Transformer (**T5**)
  - Encoder-Decoder

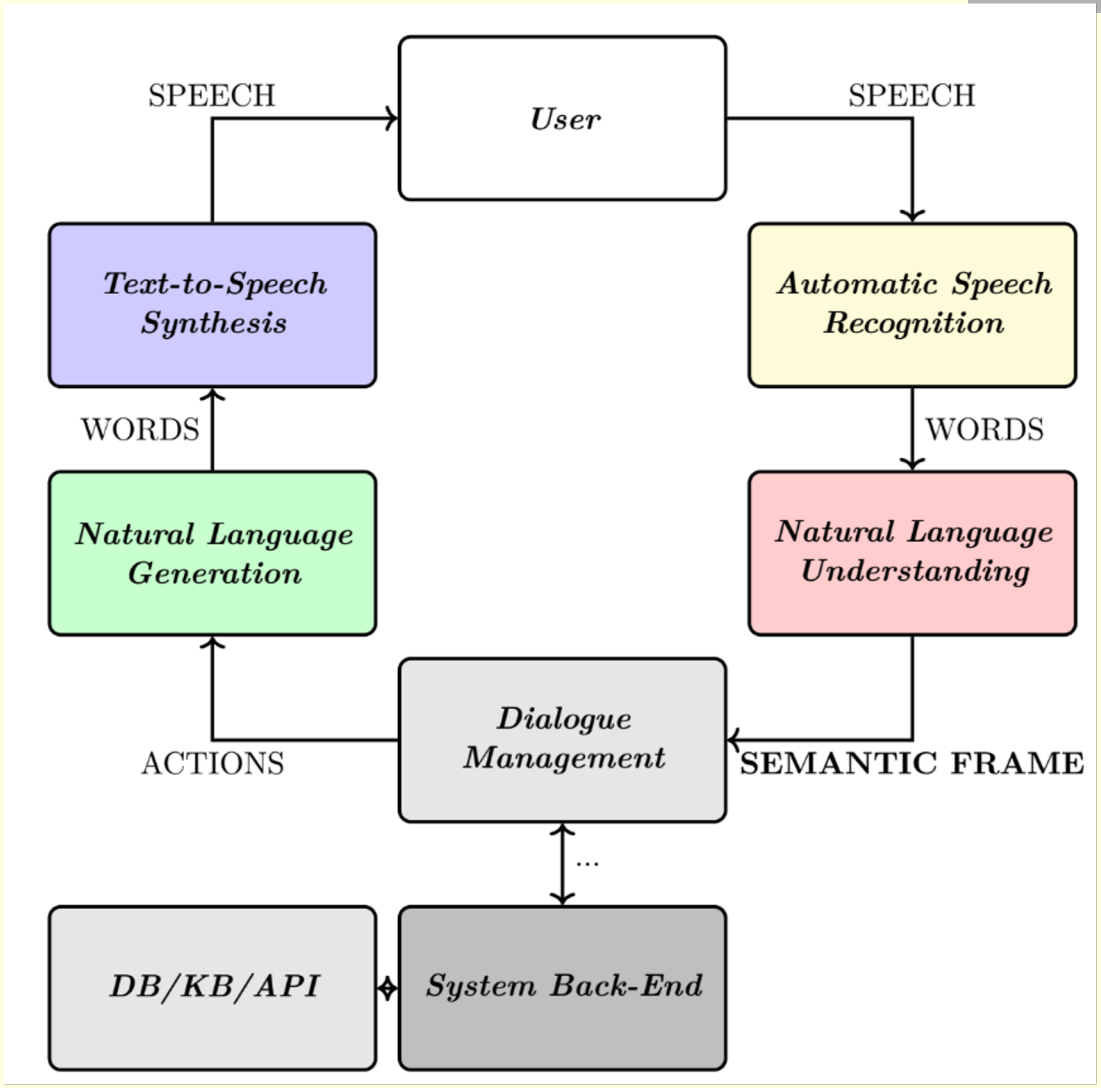
BERT

- Next Sentence Prediction
  - On [CLS] token
- Mask LM
  - [MASK] on a random token of the two sentences



Natural Language Understanding

Spoken Dialogue System



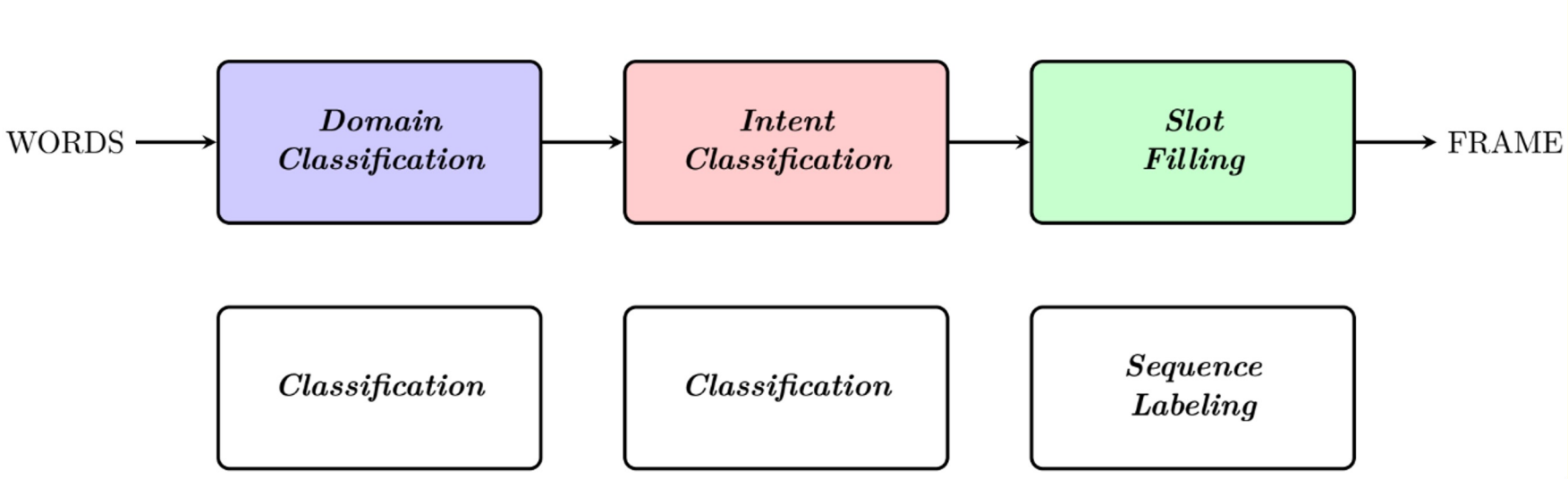
Semantic Frame

- Requires a domain ontology
  - Set of intents
  - Set of associated slots

show me horror movies based on Stephen King

find_movie(*genre*=‘horror’, *writer*=‘Stephen King’)

NLU Task


WORDS
Domain Classification
Intent Classification
Slot Filling
FRAME
Classification
Classification
Sequence Labeling

Benchmark datasets

- ATIS (Airline Travel Information System):
  - Manual transcriptions about humans asking for flight information

- SNIPS:
  - Crowdsourced queries distributed on different intents (PlayMusic, RateBook etc.)

Intent Classification and Slot filling

- **Intent Classification**:
  - Text Classification task
    - Models: MultinomialNB, SVM
- **Slot Filling**:
  - Shallow parsing task (IOB tags)
  - Models: CRF, HMM

Intent Classification and Slot filling

INTER-DEPENDENCY !

- **Intents**:
  - An intent has its own specific slots and values
- **Slots**:
  - Some slots and values are specific to an intent

Intent: find_book -> {author, genre, date..}

Slots: {departure, terminal..} -> book_fligth

Multi-task Learning

MTL is to train a single model on multiple related tasks to achieve inductive transfer between the tasks.

Inductive transfer enhances the generalisation by introducing an additional source of inductive bias for the learner to prefer a hypothesis over the others.

Caruana, R. (1998). Multitask Learning. In: Thrun, S., Pratt, L. (eds) Learning to Learn. Springer, Boston, MA.

Multi-task Learning (RNN)

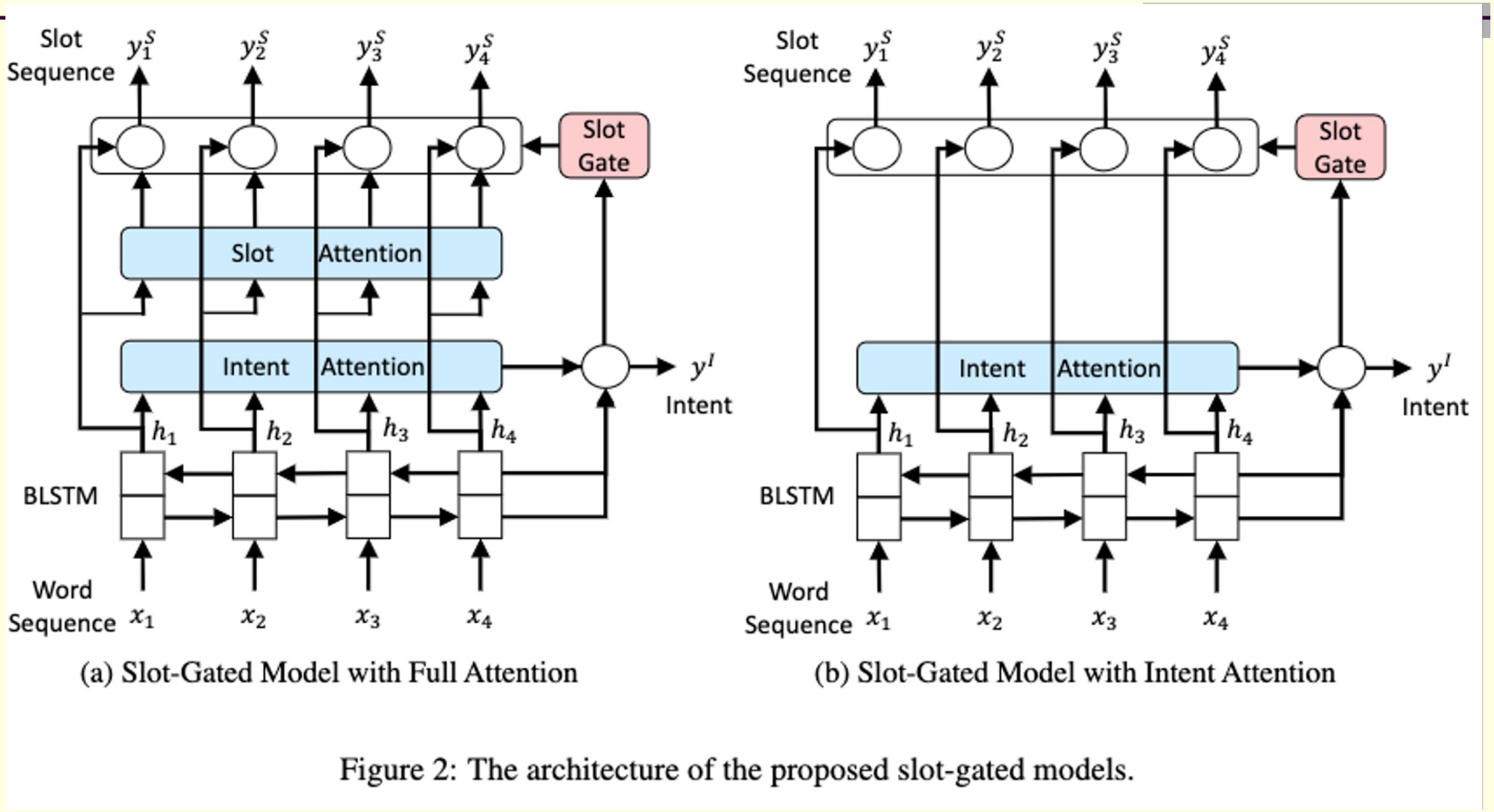


Figure 2: The architecture of the proposed slot-gated models.

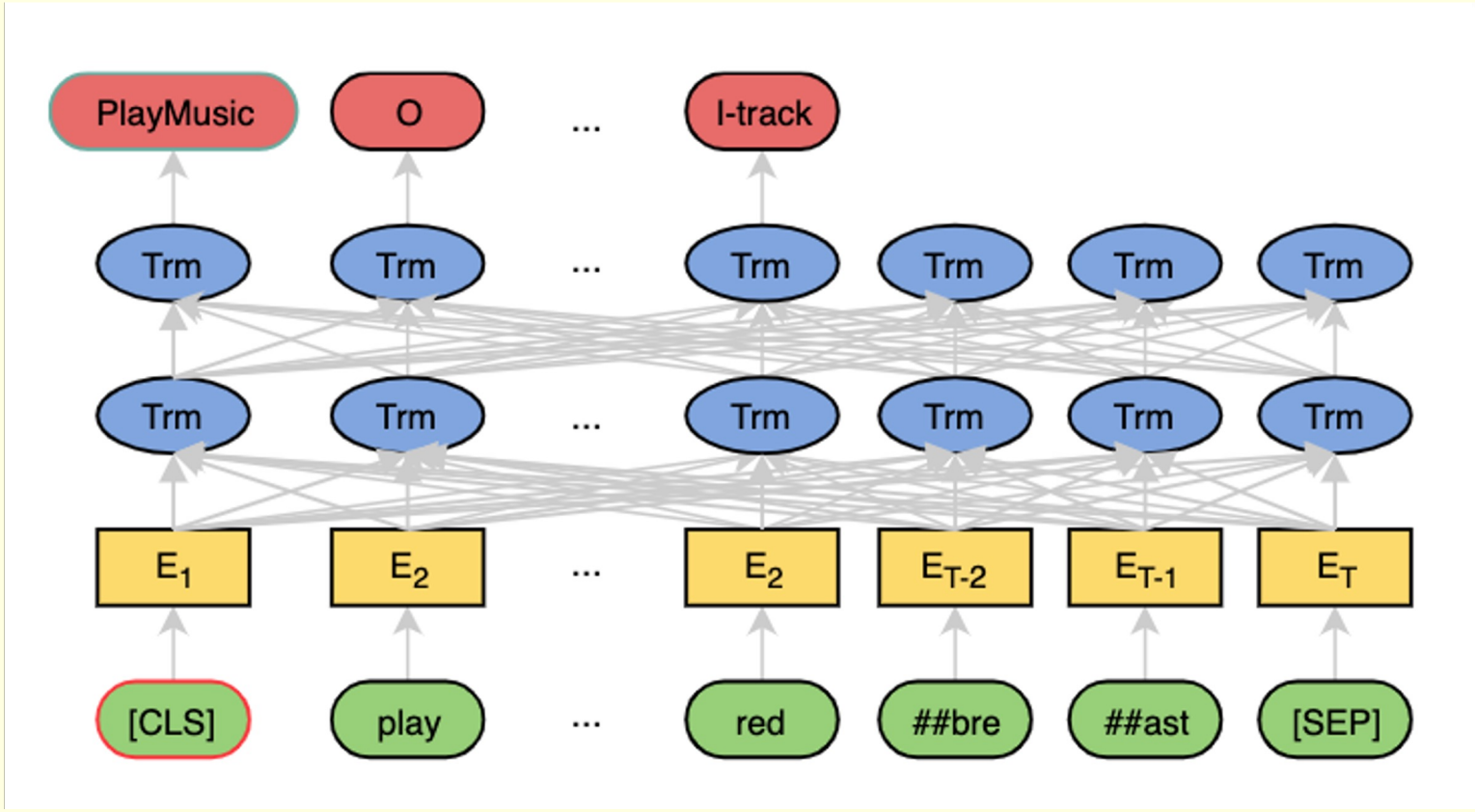
Slot-Gated Modeling for Joint Slot Filling and Intent Prediction (Goo et al., NAACL 2018)

Multi-task Learning (RNN)

Model		ATIS Dataset			Snips Dataset		
		Slot (F1)	Intent (Acc)	Sentence (Acc)	Slot (F1)	Intent (Acc)	Sentence (Acc)
Joint Seq. (Hakkani-Tür et al., 2016)		94.3	92.6	80.7	87.3	96.9	73.2
Atten.-Based (Liu and Lane, 2016)		94.2	91.1	78.9	87.8	96.7	74.1
Proposed	Slot-Gated (Full Atten.)	94.8†	93.6†	82.2†	**88.8**†	**97.0**	**75.5**†
	Slot-Gated (Intent Atten.)	**95.2**†	**94.1**†	**82.6**†	88.3	96.8	74.6

Slot-Gated Modeling for Joint Slot Filling and Intent Prediction (Goo et al., NAACL 2018)

Multi-task Learning (Transformer)



Chen, Qian, Zhu Zhuo, and Wen Wang. "Bert for joint intent classification and slot filling." *arXiv preprint arXiv:1902.10909* (2019)

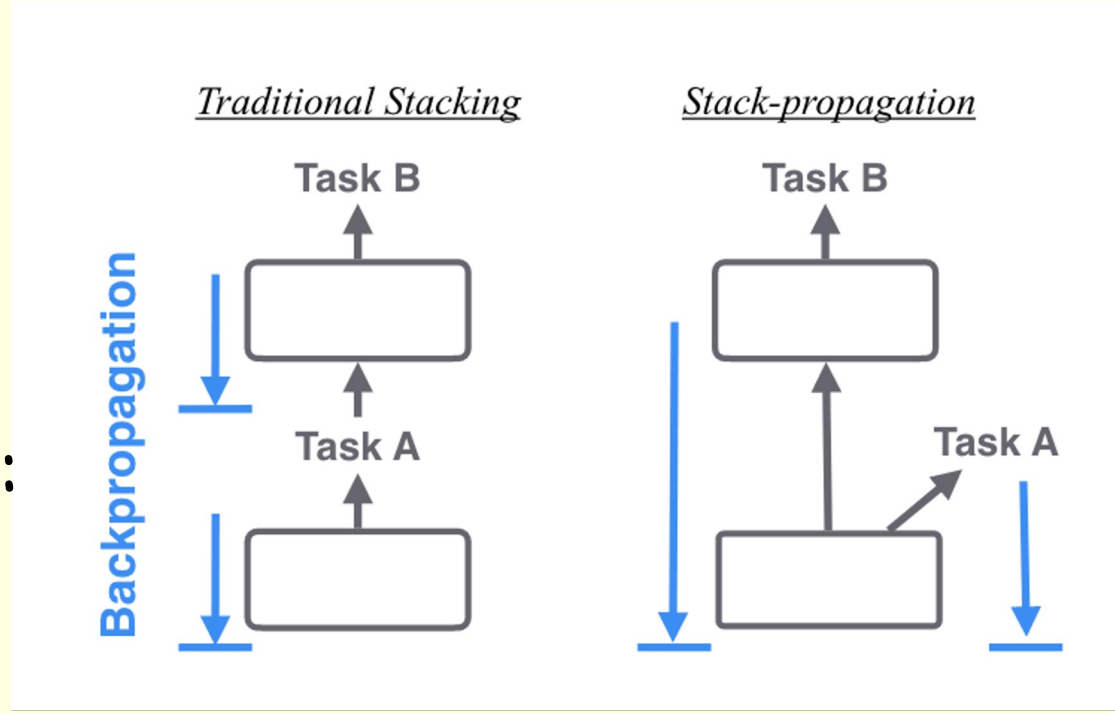
Multi-task Learning (Transformer)

Models	Snips			ATIS		
	Intent	Slot	Sent	Intent	Slot	Sent
RNN-LSTM (Hakkani-Tür et al., 2016)	96.9	87.3	73.2	92.6	94.3	80.7
Atten.-BiRNN (Liu and Lane, 2016)	96.7	87.8	74.1	91.1	94.2	78.9
Slot-Gated (Goo et al., 2018)	97.0	88.8	75.5	94.1	95.2	82.6
Joint BERT	**98.6**	**97.0**	**92.8**	97.5	**96.1**	88.2
Joint BERT + CRF	98.4	96.7	92.6	**97.9**	96.0	**88.6**

Chen, Qian, Zhu Zhuo, and Wen Wang. "Bert for joint intent classification and slot filling." *arXiv preprint arXiv:1902.10909* (2019)

Two-Step prediction

- Pipeline:
  - The prediction of one task is used to predict the other task
- Stack-propagation:
  - The prediction of Task A is differentiable



Zhang, Y., & Weiss, D. (2016, August). Stack-propagation: Improved Representation Learning for Syntax. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1557-1566).

Two-Step with shared layers

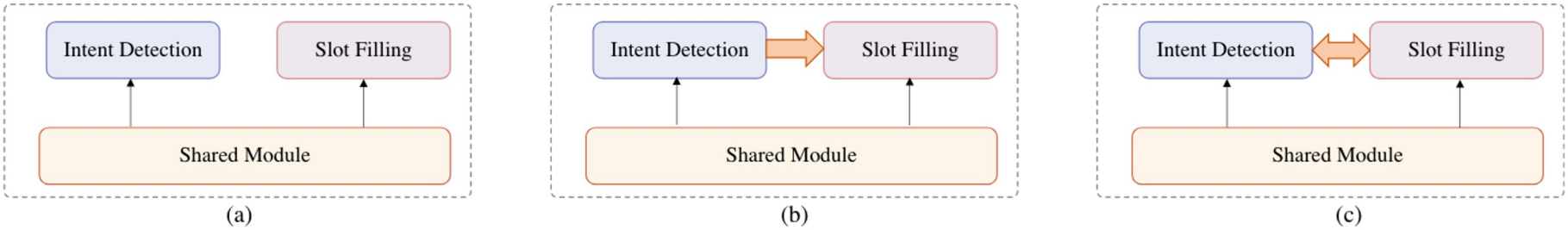
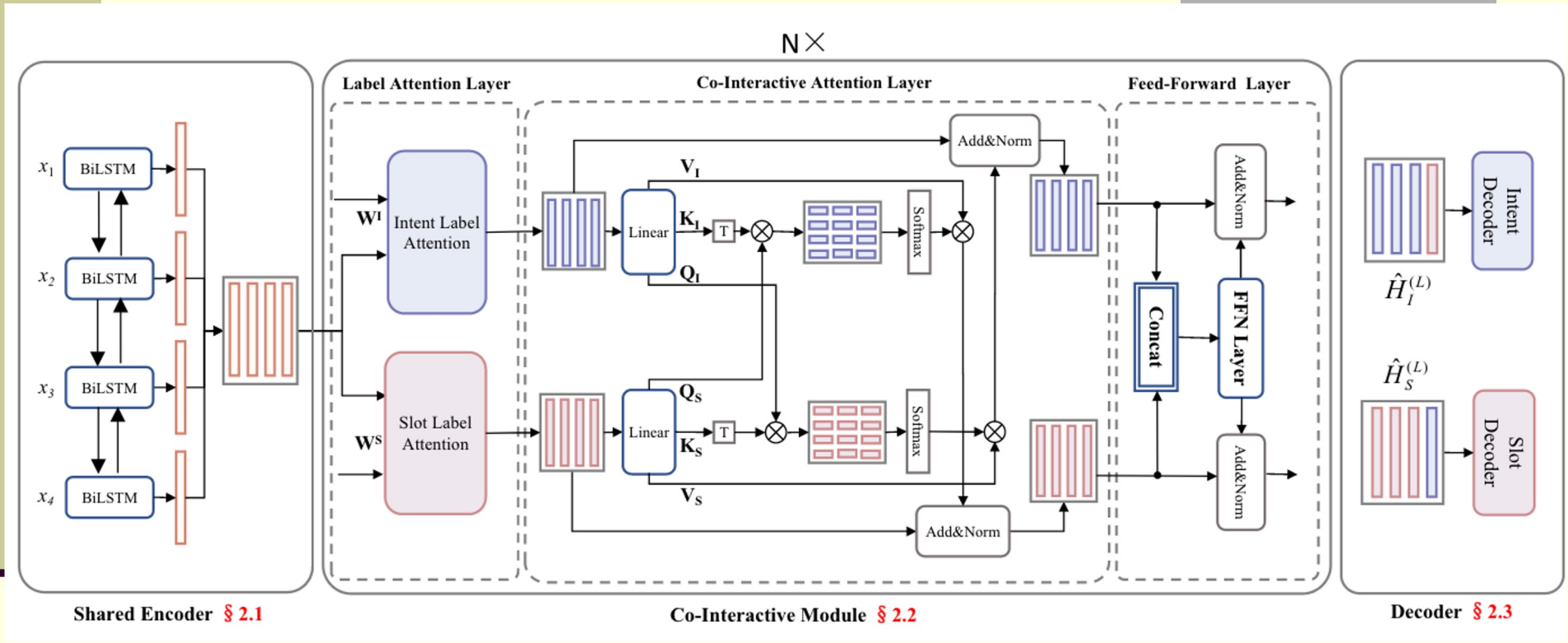


Figure 1: Methods for joint SLU tasks. (a) Multi-task framework with sharing parameters. (b) Prior work explicitly apply intent information to guide the slot filling. (c) Our proposed co-interactive Transformer to establish a bidirectional connection.

Two-Step with shared layers



Qin, Libo, et al. "A co-interactive transformer for joint slot filling and intent detection." *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021.

Two-Step with shared layers

Model	SNIPS			ATIS		
	Slot (F1)	Intent (Acc)	Overall (Acc)	Slot (F1)	Intent (Acc)	Overall (Acc)
Slot-Gated Atten (Goo et al., 2018)	88.8	97.0	75.5	94.8	93.6	82.2
Self-Attentive Model (Li et al., 2018)	90.0	97.5	81.0	95.1	96.8	82.2
Bi-Model (Wang et al., 2018)	93.5	97.2	83.8	95.5	96.4	85.7
CAPSULE-NLU (Zhang et al., 2019)	91.8	97.3	80.9	95.2	95.0	83.4
SF-ID Network (E et al., 2019)	90.5	97.0	78.4	95.6	96.6	86.0
CM-Net (Liu et al., 2019)	93.4	98.0	84.1	95.6	96.1	85.3
Stack-Propagation (Qin et al., 2019)	94.2	98.0	86.9	95.9	96.9	86.5
Our framework	**95.9***	**98.8***	**90.3***	**95.9**	**97.7***	**87.4***

Qin, Libo, et al. "A co-interactive transformer for joint slot filling and intent detection." *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021.