LLM 101

NCI Training

Zhuochen Wu

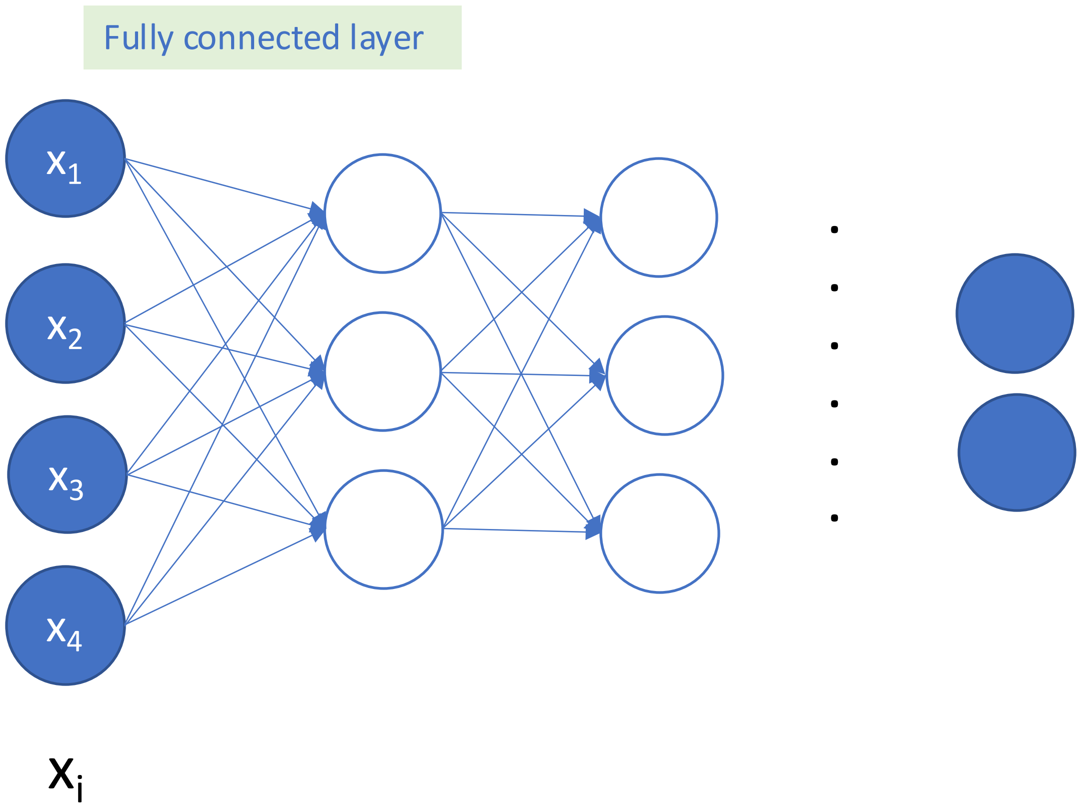
Outline

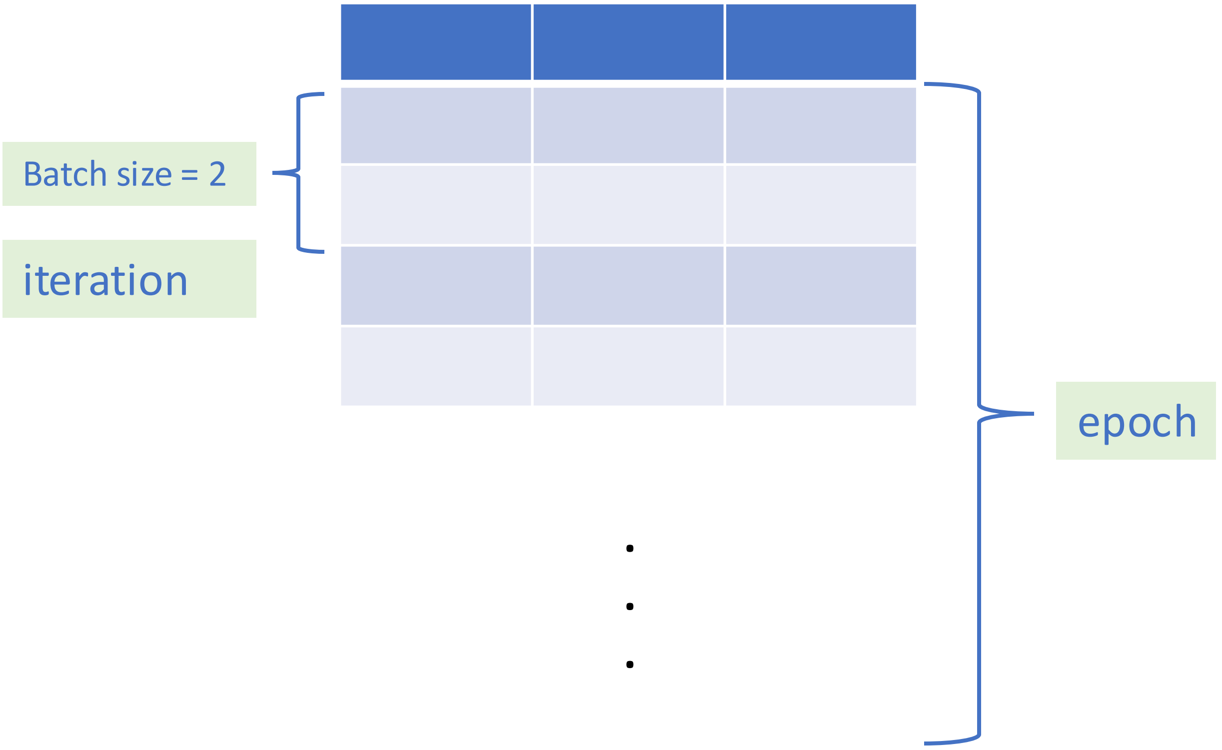
- Introduction to Machine Learning and Deep Learning
- Text Processing
- Transformers

Machine learning

Learning methods	Supervised	Unsupervised	Reinforced
Components	Dataset	Algorithm	Features
Examples	Numbers, text, images, speech…	Linear regression, Decision Tree, SVM……	Attributes in data

Deep Learning… And more data!





Text Processing

- Text cleaning
- Co-occurance
- Word2vec
  - CBOW

Data Cleaning → Tokenisation → Stopwords Removal → Lemmatization/ Stemming

- Data structure
- Data source
- Common dirty strings:
  - HTML tags
  - Human typos
  - Data encoding
  - Punctuations

➢ "A touching movie!!\n It is full of emotions and wonderful acting\n\n\n.
 I could have sat through it a second time.”

Python string functions
Regular expression – Regex

➢ “A touching movie It is full of emotions and wonderful acting I could have sat through it a second time”

Data Cleaning → Tokenisation → Stopwords Removal → Lemmatization. Stemming

- “A touching movie It is full of emotions and wonderful acting I could have sat through it a second time”

Python string functions
Libraries: Nltk, WordNet, Spacy …

- [“a”, “touching”, “movie”, “it”, "is”, “full”, “of”, “emotions”, “and”, “wonderful”, “acting”, “I”, “could”, “have”, “sat”, “through”, “it”, “a”, “second”, “time”]

Data Cleaning → Tokenisation → Stopwords Removal → Lemmatization/ Stemming

- [“a”, “touching”, “movie”, “it”, "is”, “full”, “of”, “emotions”, “and”, “wonderful”, “acting”, “I”, “could”, “have”, “sat”, “through”, “it”, “a”, “second”, “time”]

Python string functions (customization)
Libraries: Nltk, WordNet, Spacy ...

- [“touching”, “movie”, “full”, “emotions”, “wonderful”, “acting”, “have”, “sat”, “second”, “time”]

Data Cleaning → Tokenisation → Stopwords Removal → Lemmatization/ Stemming

- ["touching", "movie", "full", "emotions", "wonderful", "acting", "have", "sat", "second", "time"]

Libraries: Nltk, WordNet, Spacy …

- ["touching", "movie", "full", "emotion~~s~~", "wonderful", "act~~ing~~", "have", "s~~a~~it", "second", "time"]
- ["touch~~ing~~", "movi~~e~~", "full", "emot~~ions~~", "wonder~~ful~~", "act~~ing~~", "have", "s~~a~~it", "second", "time"]

NCI AUSTRALIA

How to represent words so that computer can understand?

“stand on his head”

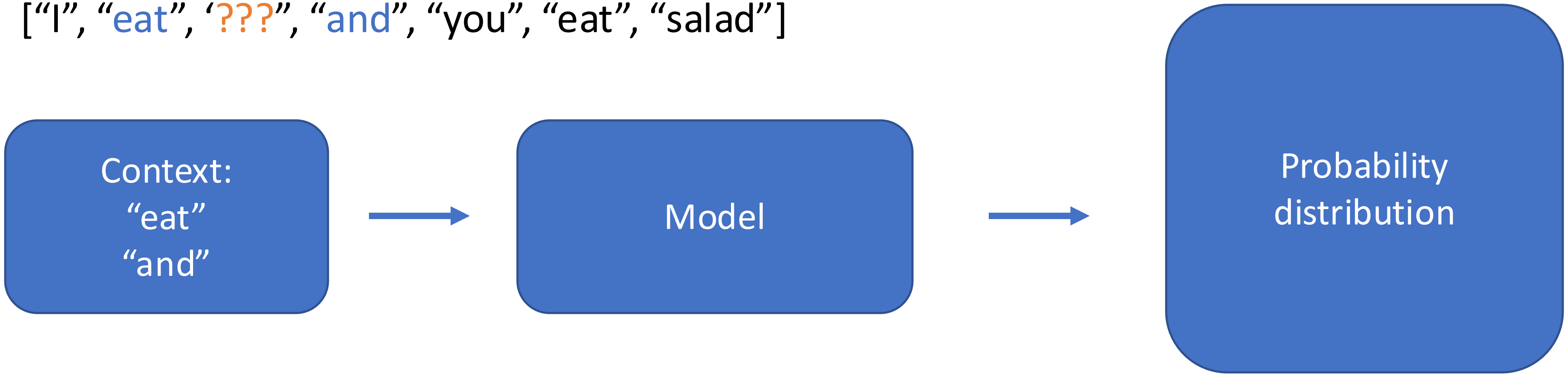
“easily”

car ⟷ **auto** **automobile** **motorcar**

- Thesaurus
- Co-occurrence
- **word2vec**
- **CBOW**

Word2vec – a prediction problem

[“I”, “eat”, ‘???”, “and”, “you”, “eat”, “salad”]



- One-hot vector:

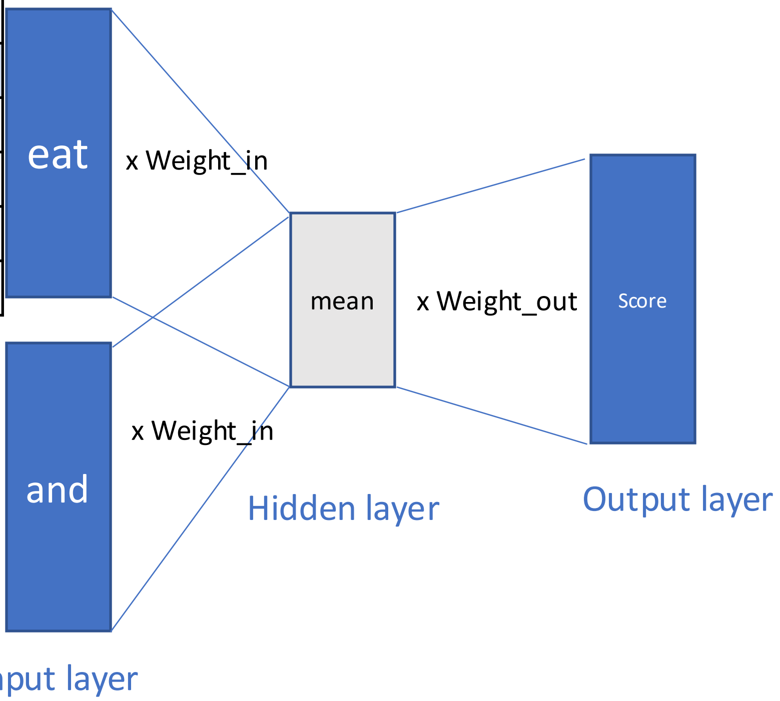
ID	Word	I	eat	burger	and	you	salad
2	eat	0	1	0	0	0	0

Word2vec

CBOW

One-hot vector

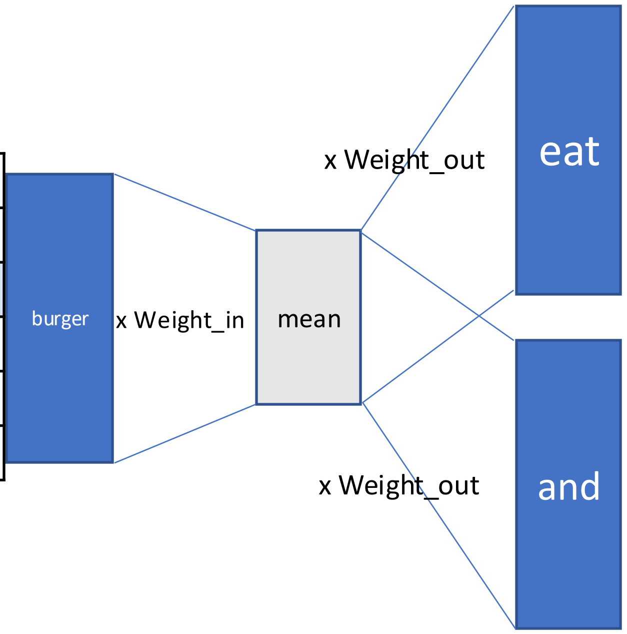
I	0
eat	1
burger	0
and	0
you	0
salad	0



Skip-gram

One-hot vector

I	0
eat	0
burger	1
and	0
you	0
salad	0



CBOW Result – Distributional Representation

- Weight_in matrix to represent words meaning
  - Syntax – plurals, past tenses…
  - Semantics
    - "king – men + women = queen"

```
[analogy] king:man = queen:?
 woman: 5.161407947540283
 veto: 4.928170680999756
 ounce: 4.689689636230469
 earthquake: 4.633471488952637
 successor: 4.6089653968811035
```

```
[analogy] take:took = go:?
 went: 4.548568248748779
 points: 4.248863220214844
 began: 4.090967178344727
 comes: 3.9805688858032227
 oct.: 3.9044761657714844
```

```
[analogy] car:cars = child:?
 children: 5.217921257019043
 average: 4.725458145141602
 yield: 4.208011627197266
 cattle: 4.18687629699707
 priced: 4.178797245025635
```

Language Model

- Language model: the probability of a sequence of words.

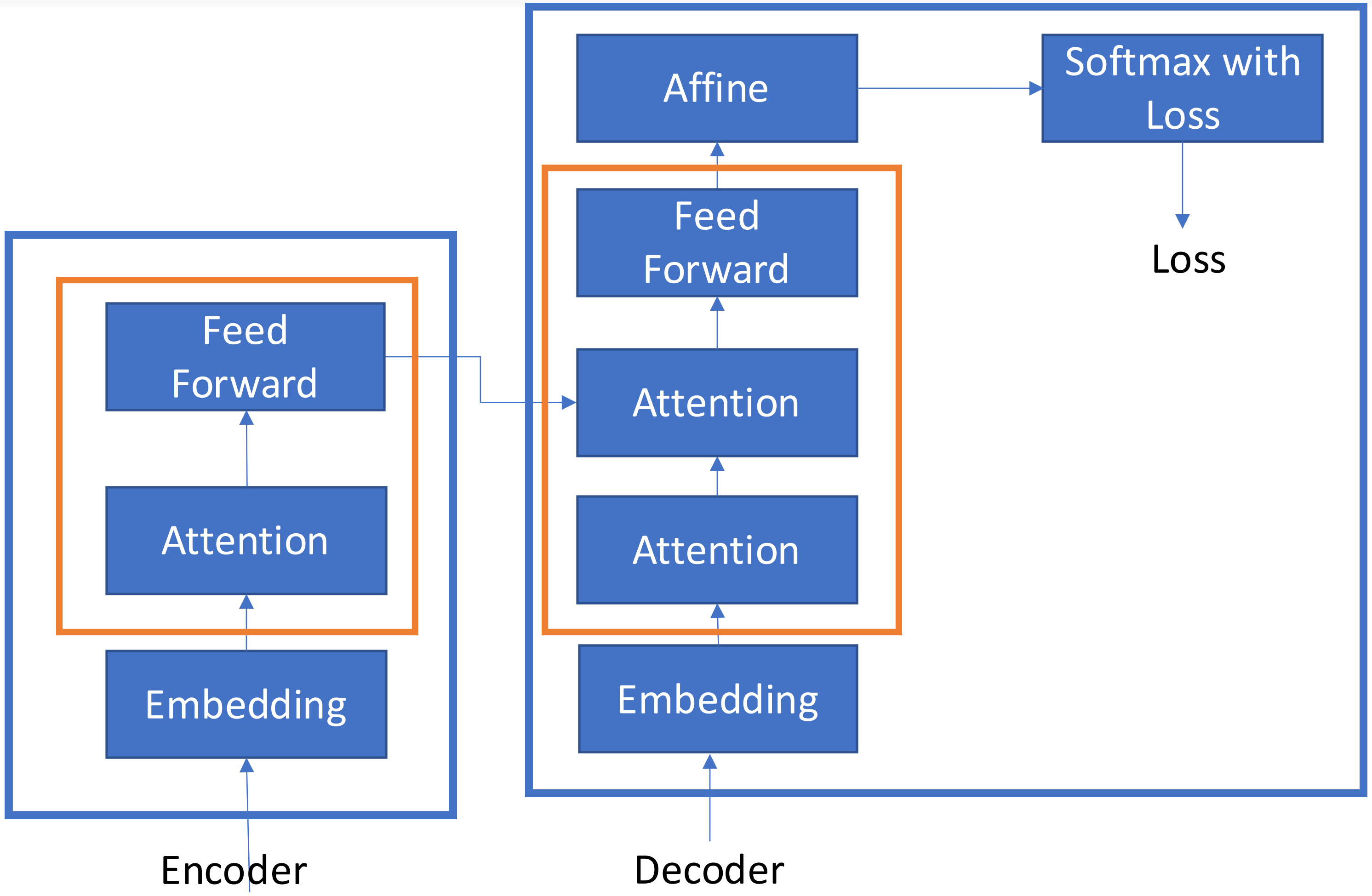
$$P(w_1, \cdots, w_m) = P(w_m \mid w_1, \cdots, w_{m-1}) P(w_{m-1} \mid w_1, \cdots, w_{m-2}) \cdots P(w_3 \mid w_1, w_2) P(w_2 \mid w_1) P(w_1) = \prod_{t=1}^{m} P(w_t \mid w_1, \cdots, w_{t-1}) \text{①}$$

$$P(A,B) = P(A|B)P(B)$$

Transformers

- Transformer
- GPT
- Bert

Transformer – Attention without RNN



- Self-attention instead of LSTM layer
- Multi-head attention

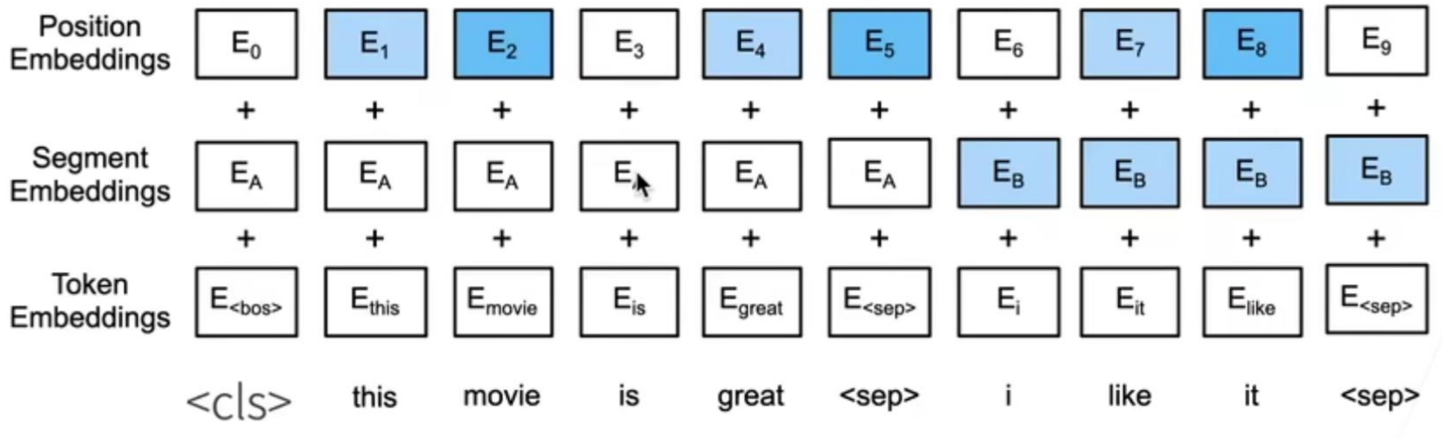
Depth repetition

Bert – NLP Model Based on Fine Tuning!

Bert – transformer without decoder

- Pre-trained model has extracted enough features
- Only need to replace output layer for a new task
- Original models in the paper:
  - Base: 12 (transformer encoder) blocks, hidden size = 768, 12 heads, 110M parameters
  - Large: 24 blocks, hidden size = 1024, 16 heads, 340M parameters
  - Trained on more than 3B words (whole Wikipedia and some books)

Bert

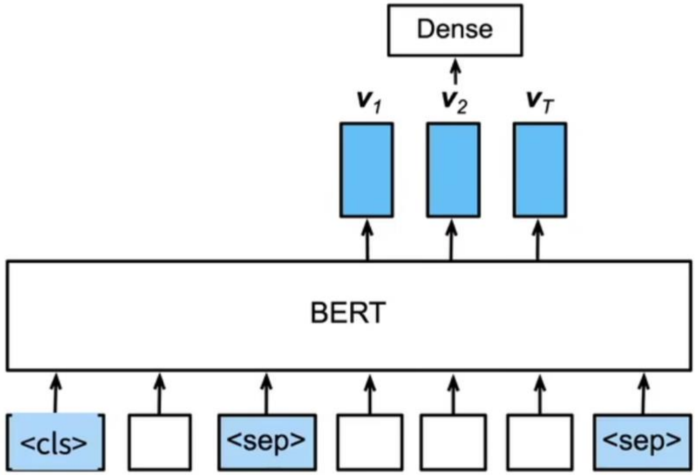


Pretraining:

- Masked language model - randomly mask some words in the sentence to predict
- Predict next sentence – 50% chance to select adjacent sentences as paired input, 50% chance to select random sentence pairs. Predict <cls> in output

1. Paired sentence input
2. Additional special tokens
3. Trainable position embedding

Bert – Q&A



Question Paragraph

Output:
- Token is the start of the answer
- Token is the end of the answer
- Neither

Fine tuning:
Increase the learning rate for output layer
Set some of the base layer parameters so finish training faster

NCI AUSTRALIA

References

- *Deep Learning from Scratch 2 © 2018 Koki Saitoh, O' Reilly Japan, Inc.*
- *Wu, Yonghui, et al. Google's neural machine translation system:Bridging the gap between human and machine translation[J]. arXiv preprint arXiv:1609.08144, 2016.*
- *https://www.youtube.com/watch?v=T05t-SqKArY&list=TLPQMjQxMDIwMjLirqQmTCjo-w&index=1*
- *https://www.youtube.com/watch?v=6ArSys5qHAU*

Questions ?

NCI Contacts

General enquiries: +61 2 6125 9800
Media enquiries: +61 2 6125 4389

Support: help@nci.org.au

Email: zhuochen.wu@anu.edu.au

Address

NCI, ANU Building 143
143 Ward Road
The Australian National University
Canberra ACT 2601