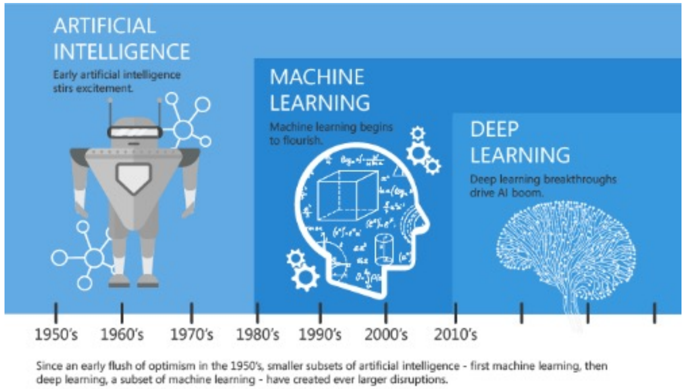
Text classification
for take-home midterm exam preparation

(Common) Text classification pipeline

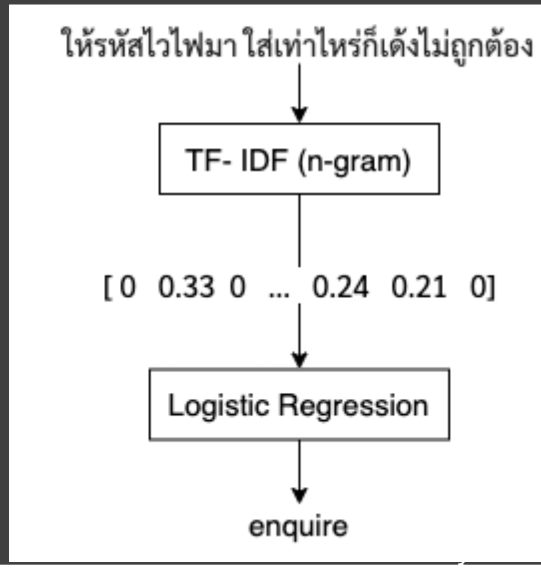
1) Data preparation → 2) Document representation → 3) Supervised learning model

Comments	Good	Like	Hate	Sentiment
Tweet1	7	8	0	😁
Tweet2	1	0	10	😡
Tweet3	2	9	1	😁



Part1: Traditional Approach

TF-IDF + Classifier



Sparse representation: Term Frequency (TF)

- Each row represents a word in the vocabulary and term-document matrix
- Each column represents a document.

vocabulary

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	5	117	0	0

document

Figure 15.1 The term-document matrix for four words in four Shakespeare plays. Each cell contains the number of times the (row) word occurs in the (column) document.

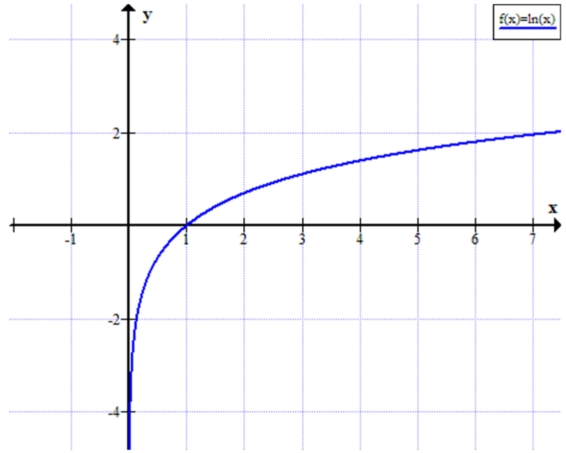
Reference: Jurafsky, Dan, and James H. Martin. Speech and language processing. 3rd edition draft, https://web.stanford.edu/~jurafsky/slp3/, August 2017

Sparse representation: TF-IDF

Need for normalization in TF

Doc1
cat = 5/10

Doc2
cat = 50/1000



- Term Frequency (TF) – per each document

$$TF(w) = \frac{\text{Frequency of word } w \text{ in } \boxed{\text{a document}}}{\text{Total number of words in } \boxed{\text{the document}}}$$

- Inverse Document Frequency (IDF) – per corpus (all documents)

$$IDF(w) = \log_e\left(\frac{\text{Total number of documents}}{\text{Number of documents that contain word } w}\right)$$

penalty score
i.e., a, an, the

- TF-IDF

$$TFIDF(w) = TF(w) * IDF(w)$$

```
# TF-IDF
tfidf = TfidfVectorizer(
    ngram_range=(1,2),          # Use unigram and bigram
    tokenizer=word_tokenize,  # Use `word_tokenize` method from pythainlp for tokenizer
    min_df=2,                   # The word found less than three times in dataset is ignored
    max_df=0.9,                 # The word found more than 90% of entries is ignore
    use_idf=True,
    smooth_idf=True,
    sublinear_tf=True
)
# Logistic regresstion
model = LogisticRegression(C=4, max_iter=300, random_state=42)
```

Sparse representation: TF-IDF (cont.)

TF

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	5	117	0	0

TF-IDF

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	0.074	0	0.22	0.28
good	0	0	0	0
fool	0.019	0.021	0.0036	0.0083
wit	0.049	0.044	0.018	0.022

Reference: Jurafsky, Dan, and James H. Martin. Speech and language processing. 3rd edition draft, https://web.stanford.edu/~jurafsky/slp3/6.pdf, Feb 2020

What classifier?

- Any classifier you like
- k-NN
- Naïve Bayes
- Logistic regression
- SVM
- Neural networks

```python
# TF-IDF
tfidf = TfidfVectorizer(
    ngram_range=(1,2),          # Use unigram and bigram
    tokenizer=word_tokenize, # Use `word_tokenize` method from pythainlp for tokenizer
    min_df=2,                   # The word found less than three times in dataset is ignored
    max_df=0.9,                 # The word found more than 90% of entries is ignore
    use_idf=True,
    smooth_idf=True,
    sublinear_tf=True
)
# Logistic regresstion
model = LogisticRegression(C=4, max_iter=300, random_state=42)
```

scikit learn   Install   User Guide   API   Examples   More

Prev   Up   Next

scikit-learn 0.22.1
Other versions

Please **cite us** if you use the software.

`sklearn.linear_model.LogisticRegression`
Examples using `sklearn.linear_model.LogisticRe`

`sklearn.linear_model`.LogisticRegression

class `sklearn.linear_model.`**`LogisticRegression`**(*penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None*) [source]

Logistic Regression (aka logit, MaxEnt) classifier.

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi_class' option is set to 'ovr', and uses the cross-entropy loss if the 'multi_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag', 'saga' and 'newton-cg' solvers.)

This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag', 'saga' and 'lbfgs' solvers. **Note that regularization is applied by default**. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the 'saga' solver.

Read more in the User Guide.

Parameters: **penalty : *{'l1', 'l2', 'elasticnet', 'none'}, default='l2'***

Used to specify the norm used in the penalization. The 'newton-cg', 'sag' and 'lbfgs' solvers support only l2 penalties. 'elasticnet' is only supported by the 'saga' solver. If 'none' (not supported by the liblinear solver), no regularization is applied.

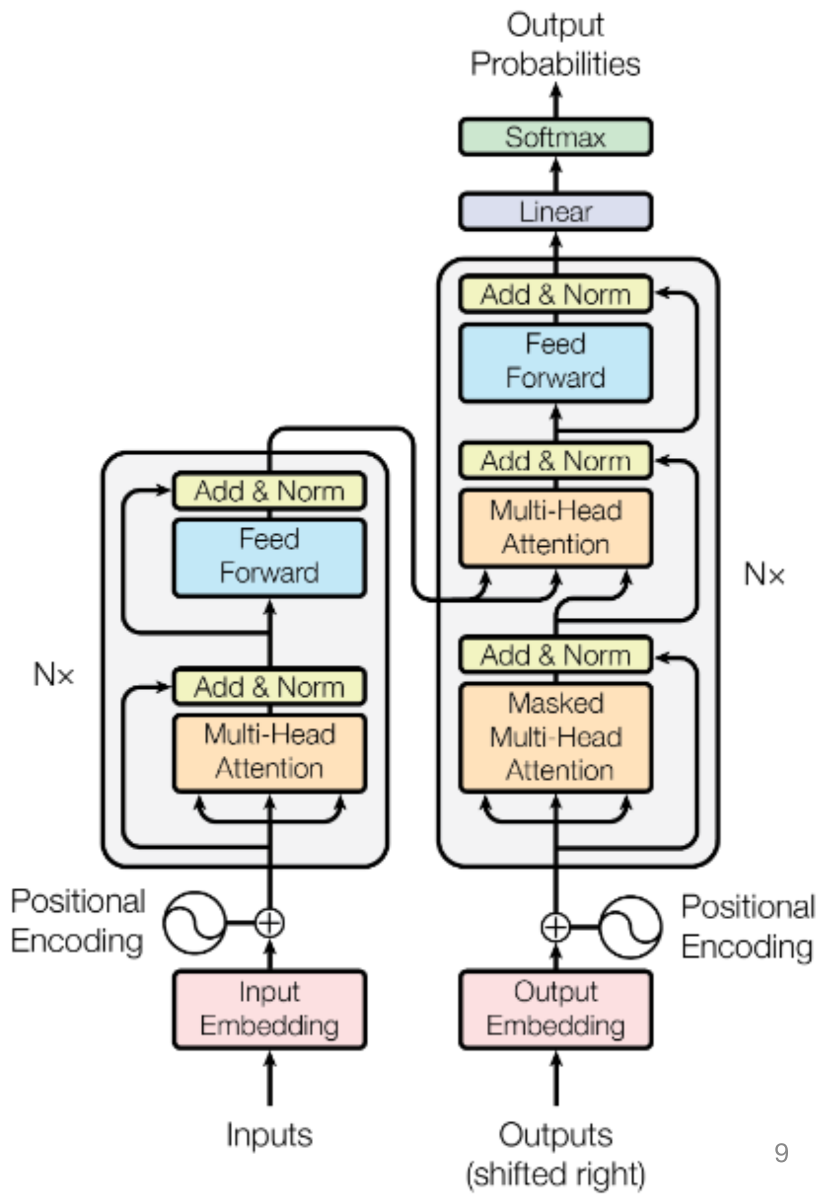
New in version 0.19: l1 penalty with SAGA solver (allowing 'multinomial' + L1)

Part2: Transformer-based models

Transformer-based models

Transformer

- A model based on attention mechanism
  - Gaining popularity in many application domain (NLP, speech, vision, bioinformatics, Reinforcement learning, Recommendation systems, etc.)



Attention is all you need 2017 https://arxiv.org/abs/1706.03762

Dall E

https://openai.com/blog/dall-e/

(a) a tapir made of accordion. a tapir with the texture of an accordion.

(b) an illustration of a baby hedgehog in a christmas sweater walking a dog

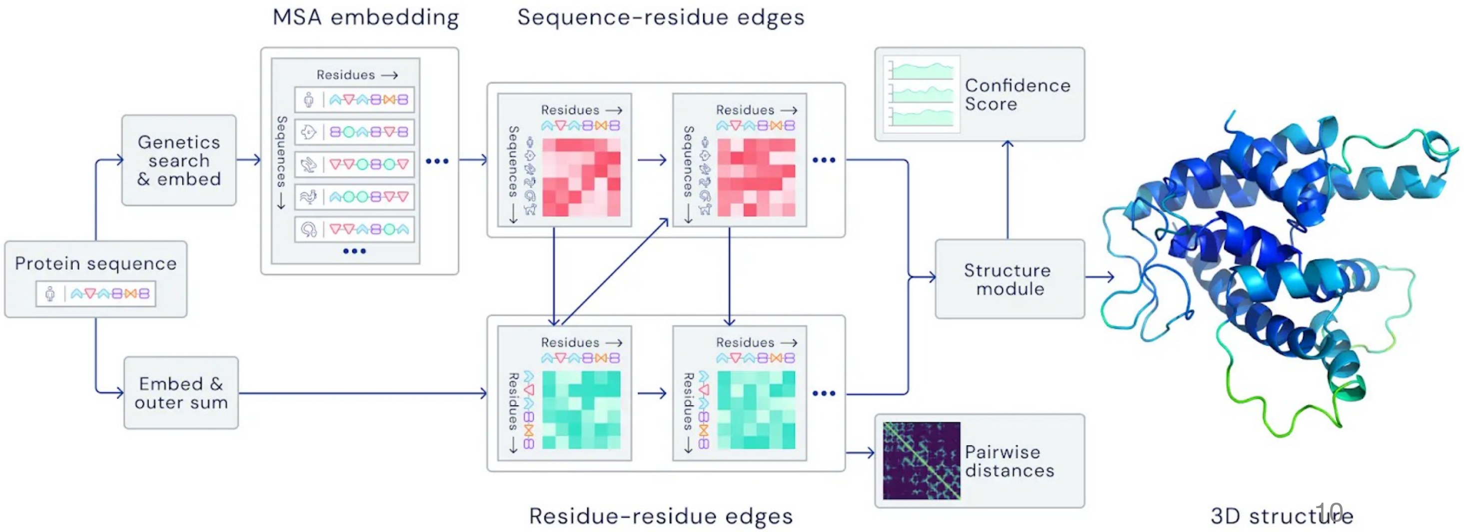
(c) a neon sign that reads "backprop". a neon sign that reads "backprop". backprop neon sign

(d) the exact same cat on the top as a sketch on the bottom

Figure 2. With varying degrees of reliability, our model appears to be able to combine distinct concepts in plausible ways, create anthropomorphized versions of animals, render text, and perform some types of image-to-image translation.

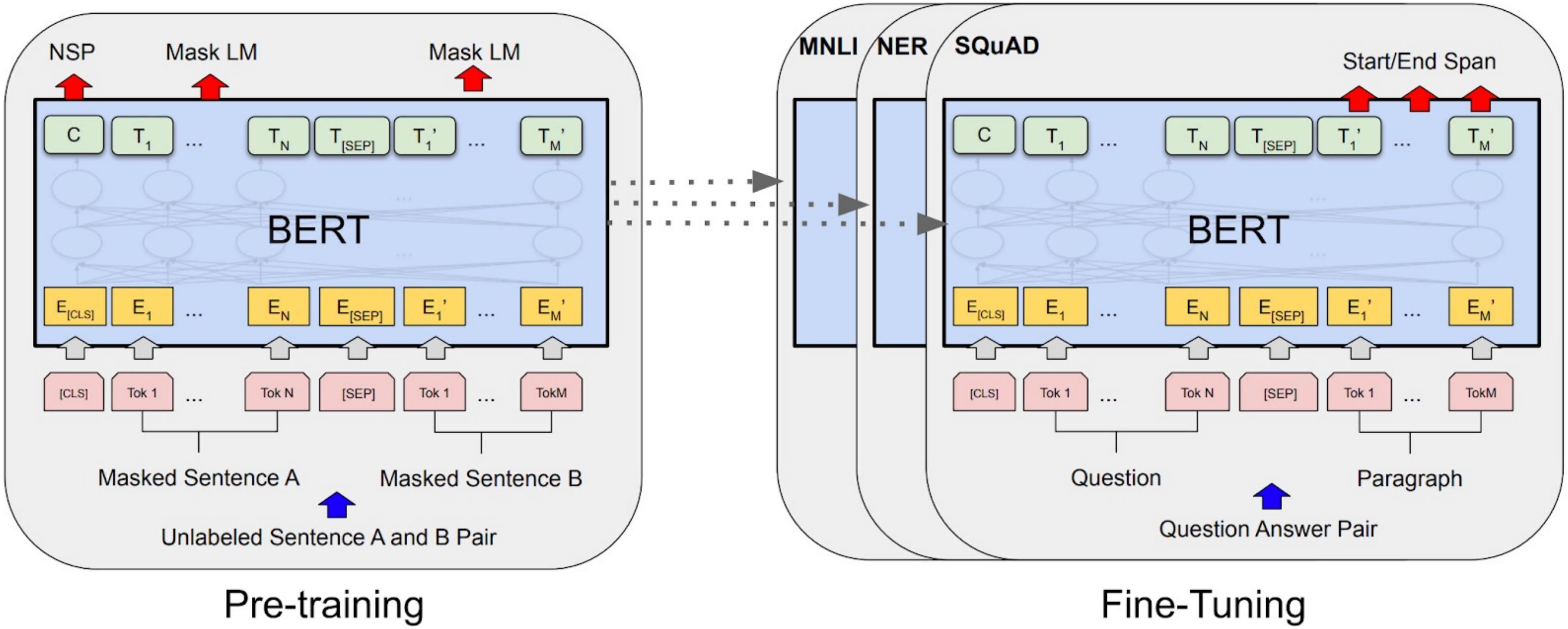
AlpheFold2

https://www.nature.com/articles/d41586-020-03348-4



BERT

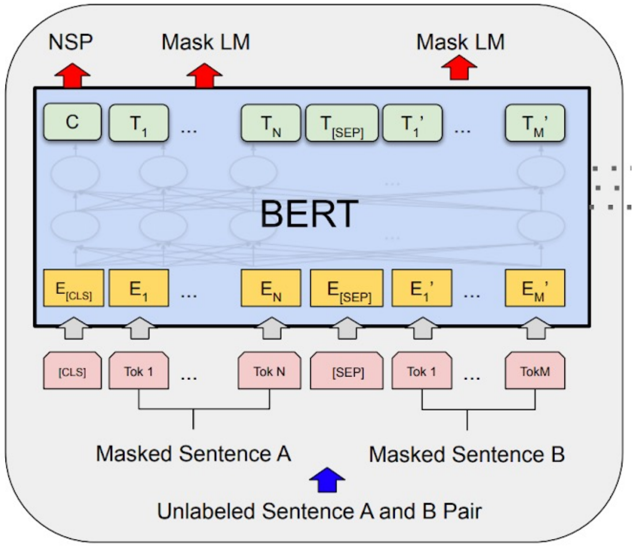
- Pretrained language model based on transformers
  - Can be used in many NLP tasks



BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018 https://arxiv.org/abs/1810.04805

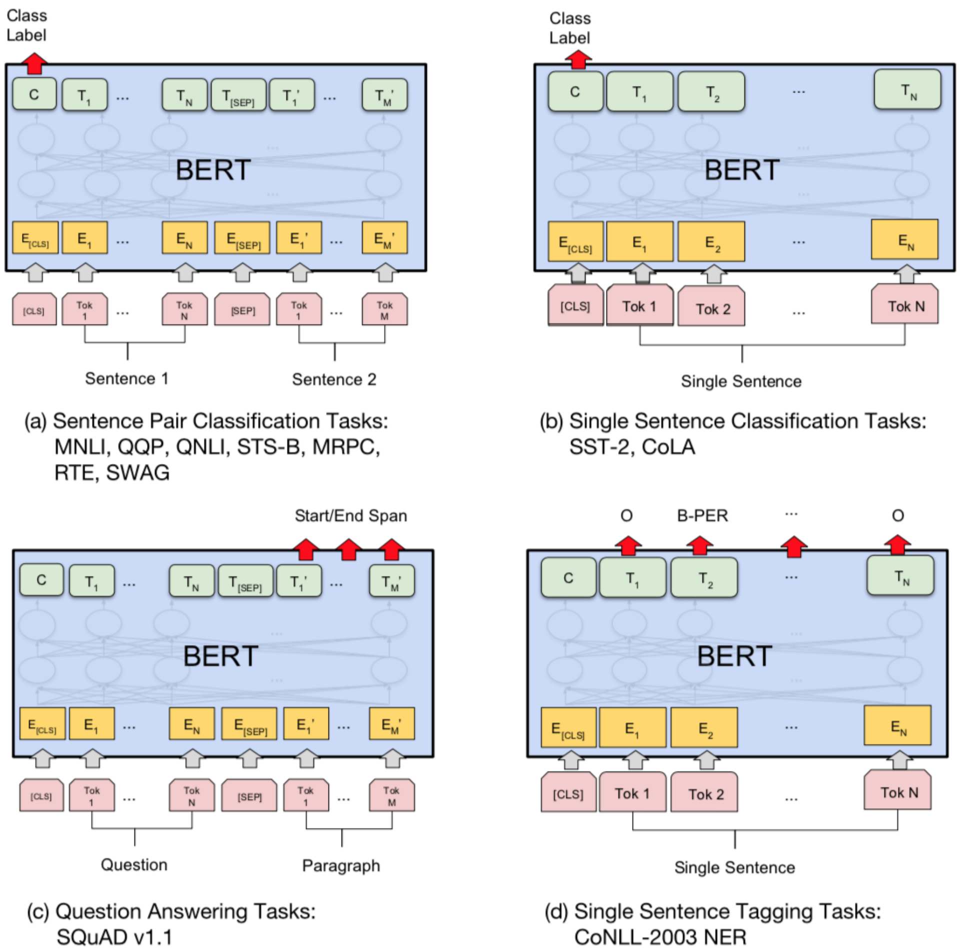
Pre-training BERT

- Predicting masked words in a sentence
  - The quick brown fox jumps over the [MASK]
  - Variants: predict correct word or not, predict swapped words, etc.
- Next sentence prediction
  - A: The cat is scared. B: It hides under the table.
  - A: The apple is on the table. B: It always rain.
  - Variants: sentence order prediction



Pre-training

Downstream tasks with BERT



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

ROBERTA (Robustly optimized BERT approach)

- A trick and tuning study
- Dynamic masking > static
- Next sentence prediction is removed

Masking	SQuAD 2.0	MNLI-m	SST-2
reference	76.3	84.3	92.8
Our reimplementation:			
static	78.3	84.3	92.5
dynamic	78.7	84.0	92.9

bsz	steps	lr	ppl	MNLI-m	SST-2
256	1M	1e-4	3.99	84.7	92.7
2K	125K	7e-4	**3.68**	**85.2**	**92.9**
8K	31K	1e-3	3.77	84.6	92.8

RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019 https://arxiv.org/abs/1907.11692

Current BERTrends

- Transformers are notorious for requiring large resources
- Newer models focus on
  - Better size/compute
  - Longer context

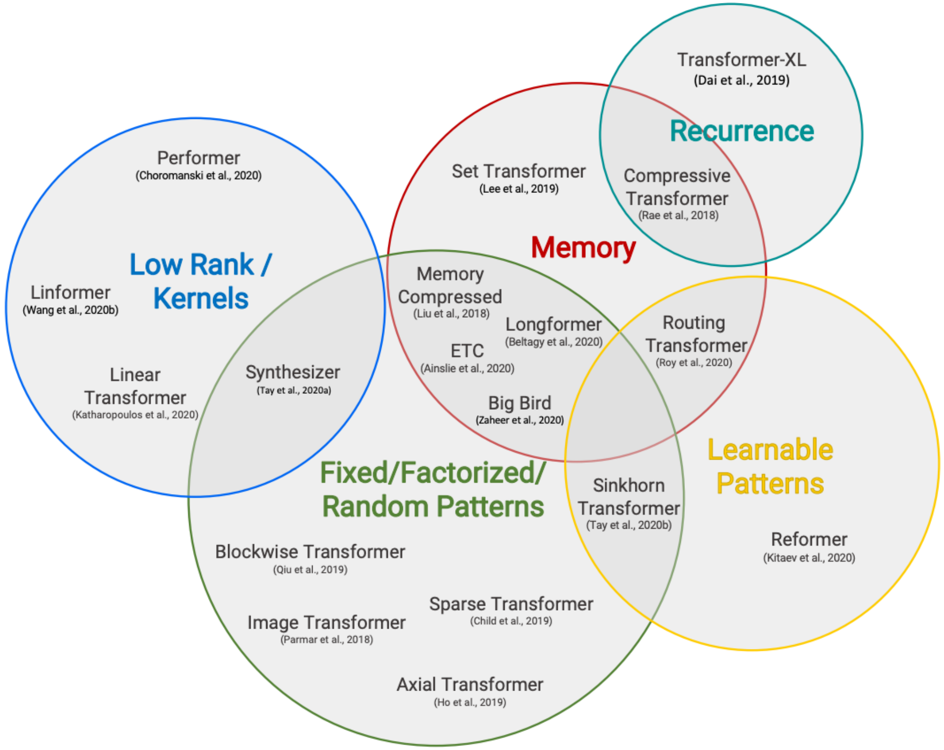
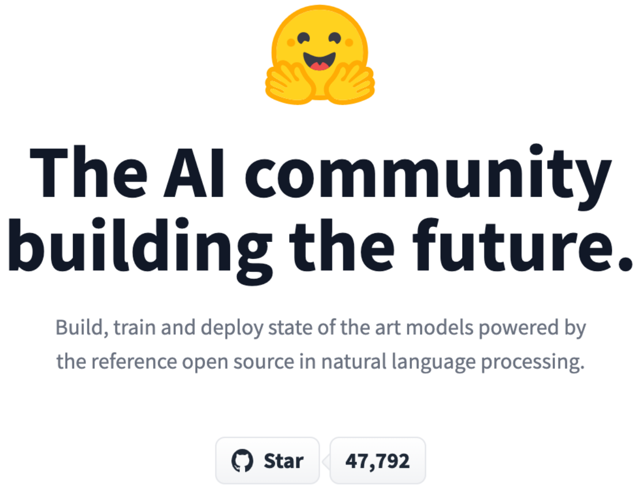


Figure 2: Taxonomy of Efficient Transformer Architectures.

Efficient Transformers: A Survey, 2020 https://arxiv.org/abs/2009.06732

Huggingface

- An opensource library for transformer-related models
- Has datasets, models, scripts, deployment solutions
- New official online course https://huggingface.co/course/chapter1



Thai pre-trained BERTs

- WangchanBERTa
- XLM-RoBERTa
- MT5

https://huggingface.co/airesearch/wangchanberta-base-att-spm-uncased
https://huggingface.co/transformers/model_doc/xlmroberta.html
https://huggingface.co/transformers/model_doc/mt5.html

Appendix

Additional resources

- NLP course @ Chula 2023 version
  - https://www.youtube.com/playlist?list=PLcBOyD1N1T-OCVsj7sGMcMd8VQCNtk50i
  - https://github.com/ekapolc/NLP_2023