Natural Language Understanding, Generation and Machine Translation - Week 5 - Prompting and Evaluating Machine Translation

Antonio León Villares

February 2023

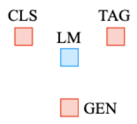
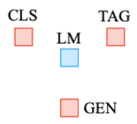
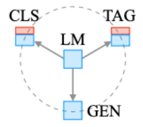
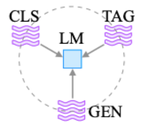
Contents

Based on:

- *Sections 1,2 and 3 of Pre-Train, Prompt and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing, by Liu et al.*
- *BLEU: A Method for Automatic Evaluation of Machine Translation*
- *C5W3L06 Bleu Score (Optional) by Andrew Ng*

1 The Paradigm of Pre-Trained Language Models

As we saw last time, ***BERT*** *lead to a* ***paradigm shift*** *in the NLP community. But this hasn't been the first such shift, nor has it been the last:*

Paradigm	Engineering	Task Relation
a. Fully Supervised Learning (Non-Neural Network)	Features (e.g. word identity, part-of-speech, sentence length)	
b. Fully Supervised Learning (Neural Network)	Architecture (e.g. convolutional, recurrent, self-attentional)	
c. Pre-train, Fine-tune	Objective (e.g. masked language modeling, next sentence prediction)	
d. Pre-train, Prompt, Predict	Prompt (e.g. cloze, prefix)	

We focus on firstly generalising and formalising the ideas behind ***pre-trained language models****, and then delve into how* ***prompting*** *has revolutionised NLP.*

1.1 Analysing Pre-Trained Language Models

- **Why are pre-trained models so powerful?**
  - there are large amounts of **naturally occurring** text data available
  - with **pre-training**, we can learn good **general** embeddings, which can be cheaply **fine-tuned** to fit specific tasks

1.1.1 Training Objectives

- **What are the main training objectives during pre-training?**

- in broad terms, **predicting** the probability of some text $\underline{x}$
- this can be done in 3 ways:
    1. **Standard Language Model**: optimise $P(\underline{x})$ based on **training corpus** (generally done by predicting tokens in a sequence one at a time).
    2. **Corrupted Text Reconstruction**: reconstruct **corrupted** text, by predicting the **de-noised** words ($P(\underline{x} \mid \tilde{\underline{x}})$, where $\tilde{\underline{x}}$ is corrupted text). The **loss** is computed solely based on the **corrupted** tokens.
    3. **Full Text Reconstruction**: reconstruct **corrupted** text, but computes the **loss** with respect to the **whole** text ($P(\underline{x} \mid \tilde{\underline{x}})$, where $\tilde{\underline{x}}$ is corrupted text).

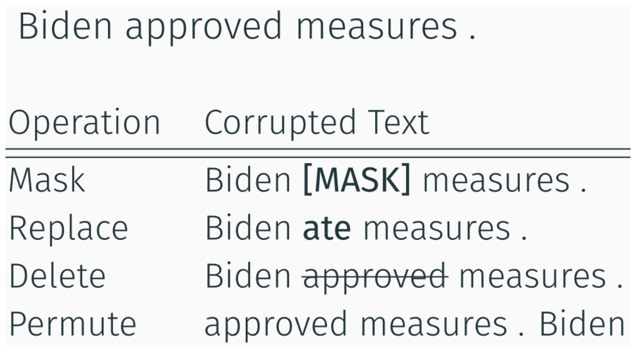
- **How does the main training objective influence the applicability of the pre-trained model?**
    - certain **downstream tasks** might benefit from different training schemes
    - for instance, **SLM** and **FTR** are better for **text generation**
    - this is particularly true for **prompting**, as we will see later on

1.1.2 Auxiliary Objectives

- **What are auxiliary objectives?**
    - beyond the **main objective** (i.e language modelling, denoising), we can pre-train models to fulfill an **auxiliary objective**
    - this allows them to **learn** language features which might be **useful** in **downstream tasks**
    - for example, in **BERT**, they use **next sentence prediction** to ensure that the embeddings are not only meaningful at the **token** level, but also at a **sentence** level (need the embeddings to correctly encode the **meaning** of a sentence if we want to be good at NSP)
- **Apart from NSP, what other auxiliary tasks are there?**
    - **Discourse Relation**: predict **rhetorical relations** between sentences (i.e is this a **question**, a **clarification**, etc...). Used by **ERNIE**.
    - **Image Region Prediction**: predict **masked regions** of an image (useful for visual-linguistic tasks)
    - also:
        * **Sentence Deshuffling**
        * **Sentence Distance Prediction**
        * **Masked Column Prediction**

1.1.3 Noising Functions

- **How can tokens be corrupted with noise during pre-training?**
    1. **Masking**: as with BERT, use a `[MASK]` token to replace tokens in the input sequence.
    2. **Replacement**: instead of replacing a token with `[MASK]`, replace with another random token.
    3. **Deletion**: directly remove a set of tokens from the input (typically used with FTR).
    4. **Permutation**: split input into **spans**, and **permute** the spans to create the new input.



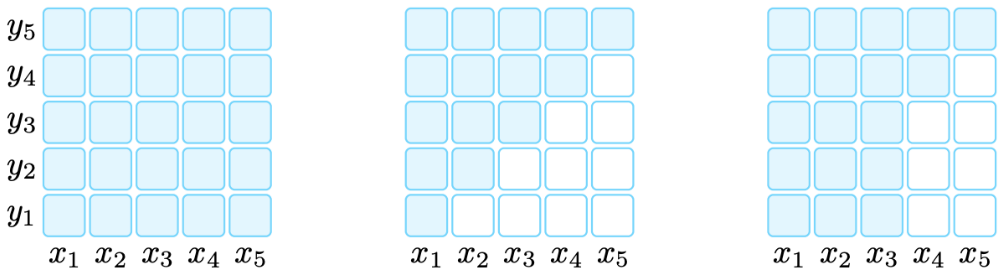
Biden approved measures .

Operation	Corrupted Text
Mask	Biden **[MASK]** measures .
Replace	Biden **ate** measures .
Delete	Biden ~~approved~~ measures .
Permute	approved measures . Biden

- **How does the type of noising affect the learning of the model?**
  - we can incorporate **prior** knowledge into the corruption
  - for example, if we want a **language model** which is sensitive to **entity prediction**, we could focus on **noisifying** entities

1.1.4 Representation Directionality

- **What is representation directionality?**
  - the way in which the input tokens are **processed** to generate **representations**
- **What are the types of directionality representation?**
  1. **Left-to-Right**: the **representation** of the **current** word depends solely on all the previous words (for example, with vanilla RNNs).
  2. **Bidirectional**: the **current representation** depends on **all** the other words in the input (for example, BERT).
  3. **Hybrid**: we can use **attention masking**, to only focus on certain parts of the input



(a) Full. (b) Diagonal. (c) Mixture.

1.1.5 Pre-Training Methods

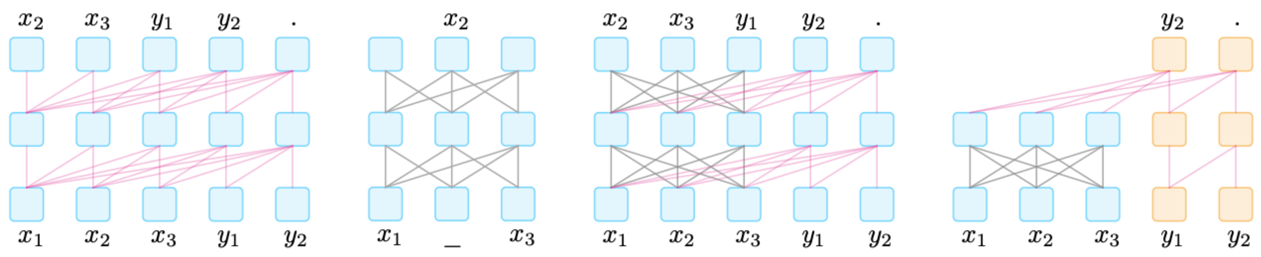
*With the notions of **training objectives**, **noising functions** and **representation directionality** in mind, we can then make choices about **how** to perform the **pre-training**.*

- **What types of pre-training strategies are there?**
  - 2 popular strategies
  - **language modelling**, which includes:
    * **Left-to-Right Language Models**
    * **Masked Language Models**

- **conditional text generation**, which includes:
  - **Prefix Language Models**
  - **Encoder-Decoder Models**

- **What are the standard language modelling architectures?**
  1. **Left-to-Right Language Models**:
     - a variety of **auto-regressive language models**, which assigns **probabilities** to **word sequences**:
       $$P(\underline{x}) = P(x_1) \times \ldots \times P(x_n \mid x_1, \ldots, x_{n-1})$$
     - this includes models like **GPT-3**
     - generally used for **prompting**
  2. **Masked Language Models**:
     - this is a **bidirectional model**, which predicts the current word given **surrounding context**:
       $$P(x_i \mid x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$$
     - by design, they're more powerful than **LtRLMs**, since these compute probabilities from left-to-right, which doesn't make them as effective for down-stream tasks, like classification)
     - examples include **BERT** and **ERNIE**
     - if used in **prompting**, better for natural language **understanding** or **analysis** (i.e classification, inference, question answering)

- **What are the conditional text generation architectures?**
  - these are models which are well-suited for **translation** and **summarisation**
  - given an **input** $\underline{x}$, seek to generate a **target** $\underline{y}$
  - this is achieved by firstly **encoding** $\underline{x}$, and then **decoding** it to generate $\underline{y}$
    1. **Prefix Language Model**:
       - $\underline{x}$ is encoded with a **fully-connected** mask (i.e full attention, the full context is attended to)
       - $\underline{y}$ is then decoded in a **left-to-right** manner
       - both the **encoding** and **decoding** are performed with the **same parameters**
       - apart from the standard language modelling objective over $\underline{y}$, might enforce a **CRT** objective on $\underline{x}$, to encourage learning good representations
    2. **Encoder-Decoder**:
       - identical to a **Prefix Language Model**, but **encoding** and **decoding** are performed with **different** parameters
       - might also enforce **CRT** objectives on $\underline{x}$
       - used in models like **T5** (more later) and **BART**



x_2 x_3 y_1 y_2 .
x_1 x_2 x_3 y_1 y_2
x_2
x_1 _ x_3
x_2 x_3 y_1 y_2 .
x_1 x_2 x_3 y_1 y_2
y_2 .
x_1 x_2 x_3 y_1 y_2

(a) Left-to-right LM. (b) Masked LM. (c) Prefix LM. (d) Encoder-Decoder.

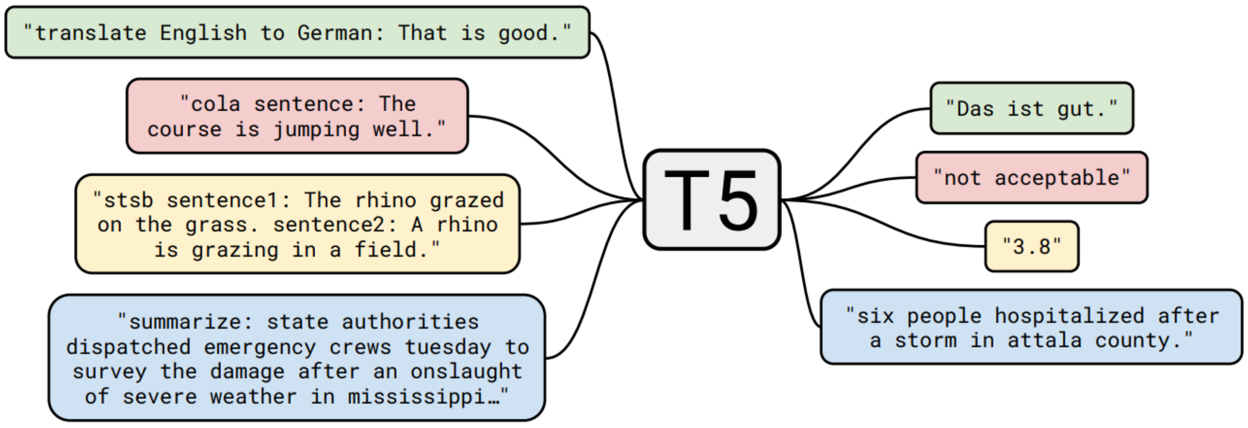
LMs	***x***			***y***			**Application**
	Mask	**Noise**	**Main Obj.**	**Mask**	**Noise**	**Main Obj.**	
L2R	Diagonal	None	SLM	-	-	-	NLU & NLG
Mask	Full	Mask	CTR	-	-	-	NLU
Prefix	Full	Any	CTR	Diagonal	None	SLM	NLU & NLG
En-De	Full	Any	None†	Diagonal	None	FTR/CRT	NLU & NLG

Figure 1: A summary of the different pre-training architectures.

1.2 Prompting

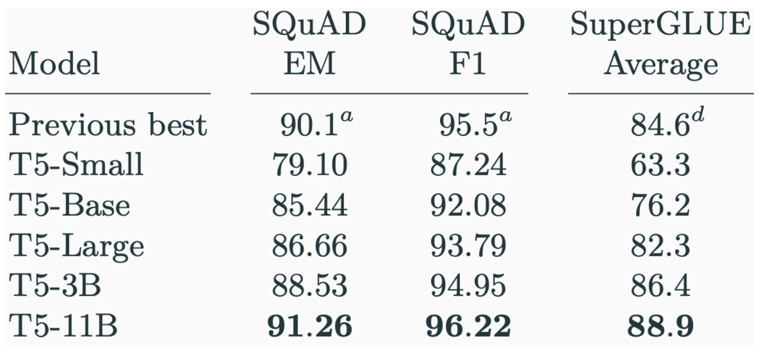
1.2.1 The T5 Language Model

- **What is the T5 language model?**
  - stands for **Text-to-Text Transfer Transformer**
  - developed by **Google** as a **unified** model for every **language** related task


"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

  - every task – including **translation**, **question answering**, and **classification** – was cast as feeding the model text as input and training it to generate some target text (see their paper)

- **How does T5 differ from BERT?**
  - unlike **BERT**, **T5** outputs **text**, not a span or an index
  - moreover, it attained **state of the art** performance when released:



Model	SQuAD EM	SQuAD F1	SuperGLUE Average
Previous best	90.1[a]	95.5[a]	84.6[d]
T5-Small	79.10	87.24	63.3
T5-Base	85.44	92.08	76.2
T5-Large	86.66	93.79	82.3
T5-3B	88.53	94.95	86.4
T5-11B	**91.26**	**96.22**	**88.9**

Figure 2: The performance of **T5**. Note that the **human performance** on SuperGLUE is 89.8 - just 0.9 greater than **T5**'s performance.

- **Why was the development of T5 important, in terms of using neural models for NLP tasks?**
  - the researchers behind **T5** tested its performance, under an exhaustive hyperparameter search
  - they found that ultimately the differentiating factor in performance is **model size** (see the table above - as the number of parameters grows to 11B, the performance keeps improving), and the **quality** of the **training data**
  - this was quite disappointing for many NLP researchers: NLP performance seemed to be capped by **size**, not **ingenuity**
  - because of this (and other reasons), **prompting** has become quite popular since 2021
- **Why does T5 exemplify one of the main challenges facing NLP?**
  - the best **T5** model required 11 **billion** parameters (as a reference, the largest **BERT** model required 340 **million**)
  - **GPT-3.5** (the basis of **ChatGPT**) required training on arouund 175 **billion** parameters
  - these sizes are **prohibitive**:
    - training is extremely costly, and has a **high carbon footprint**
    - deploying these systems is a feat of engineering, only available to large companies
  - to downsize this model, we require better **knowledge distillation**, effective techniques for **model pruning** and **quantisation** (using less accurate data types, such as 16-bit floats instead of 32-bit)

1.2.2 The Goal of Prompting

- **Intuitively, what is prompting?**
  - the **current** paradigm shift in NLP
  - since 2021:

    pre-train + fine-tune → pre-train + prompt + predict
  - with **prompts**, downstream tasks are **reformulated** to look more like the tasks solved by LMs during **pre-training**
  - in particular, the LM is tasked with "filling in the blanks" of the prompts:
    - **sentiment analysis**:

      I missed the bus today. I felt so ______.
    - **translation**:

      English: I missed the bus today. French: ___________.
- **How does prompting differ from standard supervised tasks in NLP?**
  - traditional **supervised** NLP tasks consider an input $\underline{x}$ (i.e text), and predict $\underline{y}$ (i.e text in machine translation, label in sentiment analysis, ...) by modelling:

$$P(\underline{y} \mid \underline{x}; \theta)$$

  - in **prompting**, we directly learn the probability of $\underline{x}$ itself:

$$P(\underline{x}; \theta)$$

    and use this to predict $\underline{y}$
- **Why is the prompting approach useful?**

  - with **supervised tasks**, we require a lot of data, which is often times **specific** to the task at hand (i.e predicting sentiment from movie reviews)
  - with **prompt-based** learning, we directly model language, which doesn't require large supervised datasets
  - we have plenty of **unlabelled** text data, but not enough **labelled** text data

- **How does prompting determine $\underline{y}$?**
  - this is a 3 step process:
    1. **Prompt Addition**
    2. **Answer Search**
    3. **Answer Mapping**
  - we discuss these 3 steps further below

1.2.3 Creating Prompts

- **What is prompt addition?**
  - the process of generating a **prompt**
  - for this, we use a **prompting function**, to modify the **input** $\underline{x}$:

$$\underline{x}' = f_{prompt}(\underline{x})$$

- **How is the prompting function applied?**
  - this is done in 2 steps
    1. **Template**: templates have 2 "free slots"; an **input slot** (to "hold" the input), and an **answer slot** (to "hold" the model output):

       `[X]` Overall, it was a `[Z]` movie.

    2. **Filling**: we then fill in the **input slot** with $\underline{x}$:

       $\underline{x}$ = I love this movie. $\implies$ $f_{prompt}(\underline{x})$ = I love this movie. Overall, it was a `[Z]` movie.

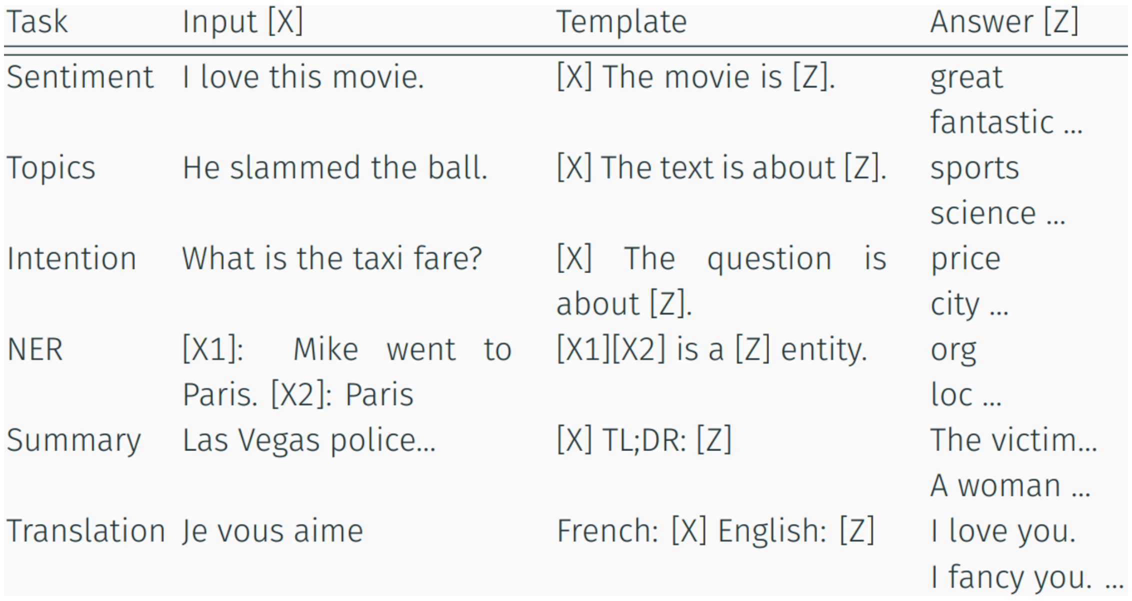
- **What is a cloze prompt?**
  - a **prompt** where the **answer slot** is in the middle

    `[X]` Overall, it was a `[Z]` movie.

- **What is a prefix prompt?**
  - a **prompt** where the **answer slot** lies at the end:

    Finnish: `[X]` English: `[Z]`

- **Can a prompt only have a single input and answer slot?**
  - no, these can be changed depending on the task at hand

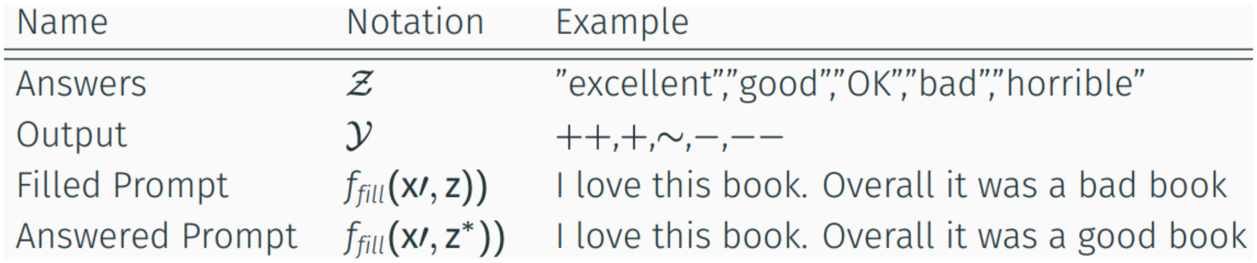


Task	Input [X]	Template	Answer [Z]
Sentiment	I love this movie.	[X] The movie is [Z].	great fantastic ...
Topics	He slammed the ball.	[X] The text is about [Z].	sports science ...
Intention	What is the taxi fare?	[X] The question is about [Z].	price city ...
NER	[X1]: Mike went to Paris. [X2]: Paris	[X1][X2] is a [Z] entity.	org loc ...
Summary	Las Vegas police...	[X] TL;DR: [Z]	The victim... A woman ...
Translation	Je vous aime	French: [X] English: [Z]	I love you. I fancy you. ...

Figure 3: Examples of prompting templates for a variety of downstream tasks.

1.2.4 Answer Search and Mapping

- **How are answers to the prompt found?**
  1. Define a set $\mathcal{Z}$ of **permissible** answers
     - for **language generation**, $\mathcal{Z}$ could be the **entire** language
     - for **classification**, $\mathcal{Z}$ could be a subset containing adjectives:
     $$\mathcal{Z} = \{excellent, good, OK, bad, horrible\}$$
  2. We define a **filling** function, which fills the **answer slot** in $\underline{x}'$ with $\underline{z} \in \mathcal{Z}$. For example, if $\underline{z} = excellent$:
     $$f_{fill}(\underline{x}', \underline{z}) = \text{I love this movie. Overall, it was an } excellent \text{ movie.}$$
  3. Using a **pre-trained** language model $P(\cdot;\theta)$, we can evaluate the best potential answer $\underline{z} \in \mathcal{Z}$:
     $$\hat{\underline{z}} = \underset{\underline{z} \in \mathcal{Z}}{search}\ P(f_{fill}(\underline{x}', \underline{z}); \theta)$$
     The *search* function could be a simple *argmax*, **beam search** or some form of **sampling** (generate outputs which follow the distribution of the LM)
- **What is answer mapping?**
  - it can be the case that the highest scoring **answer** $\hat{\underline{z}}$ isn't in the format of the desired output
  - we need some mapping to a highest scoring output $\hat{\underline{y}}$:
  $$\hat{\underline{z}} \mapsto \hat{\underline{y}}$$
  - for **language generation** tasks (i.e **translation**), this mapping can be trivial
  - for other tasks (i.e **sentiment analysis**), we need to map sets of sentiment bearing words to single classes:
  $$\{excellend, great, wonderful\} \mapsto +$$
  $$\{horrible, disgusting, bad\} \mapsto -$$



Name	Notation	Example
Answers	$\mathcal{Z}$	”excellent”,”good”,”OK”,”bad”,”horrible”
Output	$\mathcal{Y}$	++,+,∼,−,−−
Filled Prompt	$f_{fill}(x\prime, z))$	I love this book. Overall it was a bad book
Answered Prompt	$f_{fill}(x\prime, z^*))$	I love this book. Overall it was a good book

Figure 4: The full prompting paradigm for sentiment analysis: prompt addition, answer searching and answer mapping.

1.2.5 Design Considerations for Prompting

- **What is prompt engineering?**
  - the creation of **prompting functions** $f_{prompt}(\underline{x})$
  - this is a **manual task**, which isn't optimal: out of millions of prompts, how likely are we to select the **best** prompt for our task, which helps our model learn the best?
- **Can prompts be engineered automatically?**
  - if we have large corpora, they will likely contain good **prompts**
  - for example, for sentiment analysis, we can easily find corpora with sentences like "X liked Y a lot"
  - we can **sample** sentences like these, and paraphrase them to generate **prompts**
  - we can then select the prompts which result in the **highest accuracy**
- **Can prompts be continuous and learnt by the model?**
  - we can add **prompts** within the **embedding space** of the model
  - these can be **initialised** with a **discrete prompt**
  - as the model trains, we can also train the **prompt parameters**, to optimise the **template embeddings**
- **What is answer engineering?**
  - similar to **prompt engineering**, we need to make sure we have a good $\mathcal{Z}$ for each task
  - we also need to be able to correctly adapt the answer to output **mapping**

1.2.6 Prompting for Few-Shot and Zero-Shot Learning

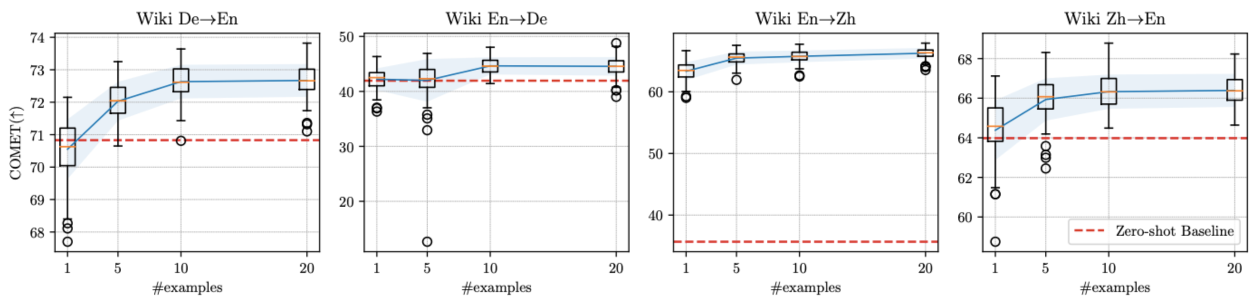
- **What is few-shot learning?**
  - a technique in which a model is **trained** to recognise new classes with only a small number of examples (often as few as one or a few examples per class)
- **What is zero-shot learning?**
  - like **few-shot learning**, but we don't provide the model with **any** example of unseen classes
  - we train the model on a smaller set of known categories, and then use additional information such as **semantic embeddings** or **attribute labels** to infer the classification of new, unseen data points
- **Is prompting effective for zero-shot learning?**

- **prompting** provides a simple framework for **zero-shot learning**
- in a recent paper by Alexandra Birch (our lecturer!), they explore this
- they create different templates for **translation**, and explore how adding line breaks affects **zero-shot learning**:

ID	Template (in English)	English		German		Chinese	
		w/o	w/	w/o	w/	w/o	w/
A	`[src]: [input] ◇ [tgt]:`	**38.78**	**31.17**	-26.15	-16.48	**14.82**	**-1.08**
B	`[input] ◇ [tgt]:`	-88.62	-85.35	-135.97	-99.65	-66.55	-85.84
C	`[input]` ◇ Translate to `[tgt]:`	-87.63	-68.75	-106.30	-73.23	-63.38	-70.91
D	`[input]` ◇ Translate from `[src]` to `[tgt]:`	-113.80	-89.16	-153.80	-130.65	-76.79	-67.71
E	`[src]: [input]` ◇ Translate to `[tgt]:`	20.81	16.69	**-24.33**	**-5.68**	-8.61	-30.38
F	`[src]: [input]` ◇ Translate from `[src]` to `[tgt]:`	-27.14	-6.88	-34.36	-9.22	-32.22	-44.95

Figure 5: Results from the Birch paper. The ◇ symbol represents the location of the line break. w/ denotes that the line break was added. The scores are **COMET** scores, which we'll see in the next section.

- **What is interesting about these results?**
  - notice, the templates are fundamentally **identical**
  - however, they obtain **drastically** different results
  - a human wouldn't struggle with this difference
- **Does few-shot learning improve the model capabilities?**
  - yes, but we see that after around 10 samples, performance stabilisies:



2 Evaluating Machine Translation

2.1 The Importance of Evaluating Machine Translation Models

- **Why is model evaluation important?**
  - many facilities nowadays rely on **well-engineered systems**
  - not being able to properly evaluate these systems can be **dangerous**
  - for example, we need to make sure that **self-driving cars** are completely safe, as malfunctioning can have serious consequences
- **Why is evaluating machine translation models important?**
  1. **Understanding System Changes**
     - having an **evaluation metric** allows us to compare different models

    - we can see whether new ideas lead to better models
    - we can verify whether new ideas change the model in an expected way
2. **Personal Implications**:
    - **machine translation** is everywhere, so we need to ensure that performance is correct
    - for example:
        * when Google switched from n-gram to neural systems, their **filters** for harmful content didn't work, since the generated output was completely different
        * when translating **menus**, understand allergies is crucial (i.e need to properly identify and translate words like "celiac")
        * multinational corporations need to be able to translate **legal requirements**, **product specifications** and **designs** to different languages

2.2 Desiderata for Machine Translation Models

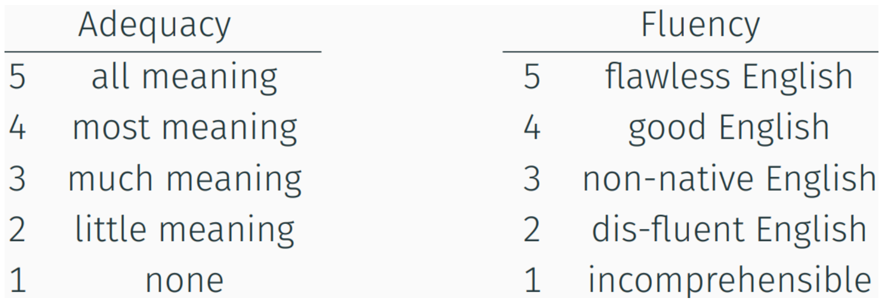
- **What are the 2 main use cases of machine translation?**

    1. **Assimilative**: understanding phrases in a different language (i.e entry requirements for a country)
    2. **Disseminative**: generating phrases in a different language (i.e translating lunch menus)

- **What 2 dimensions are crucial when evaluating machine translation?**

    1. **Adequacy**: how well does the translation convey the **meaning** of the original sentence?
    2. **Fluency**: how fluent is the output? Is it grammatical? How correct is the word choice?

    - **adequacy** and **fluency** are typically measured in a scale, from 1 to 5:

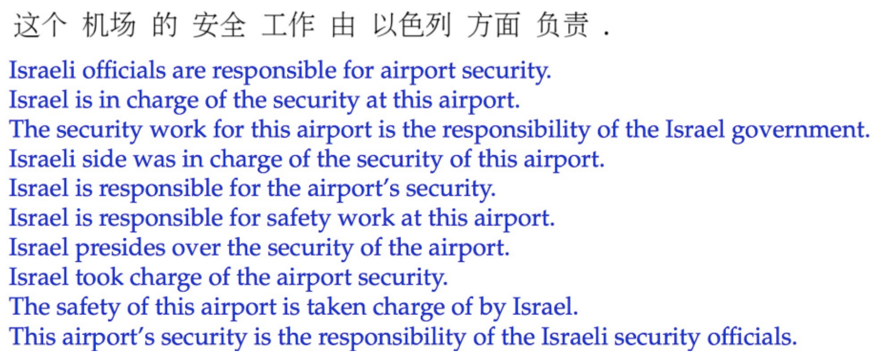


	Adequacy		Fluency
5	all meaning	5	flawless English
4	most meaning	4	good English
3	much meaning	3	non-native English
2	little meaning	2	dis-fluent English
1	none	1	incomprehensible

2.3 Human Evaluation

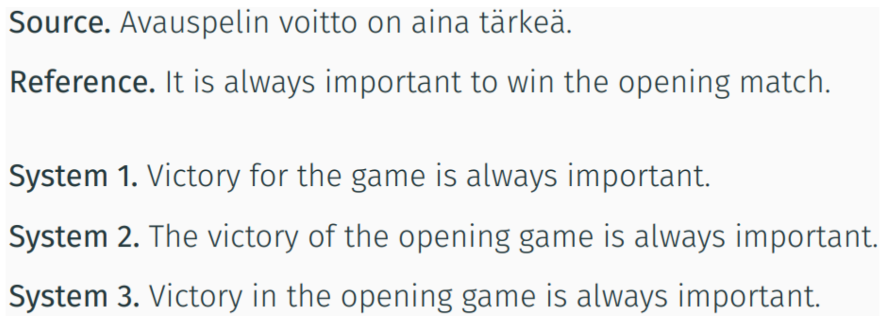
- **Why are human evaluations of translation not ideal?**

    - the **same** sentence can be **translated** in many different ways:



这个 机场 的 安全 工作 由 以色列 方面 负责 .

Israeli officials are responsible for airport security.
Israel is in charge of the security at this airport.
The security work for this airport is the responsibility of the Israel government.
Israeli side was in charge of the security of this airport.
Israel is responsible for the airport's security.
Israel is responsible for safety work at this airport.
Israel presides over the security of the airport.
Israel took charge of the airport security.
The safety of this airport is taken charge of by Israel.
This airport's security is the responsibility of the Israeli security officials.



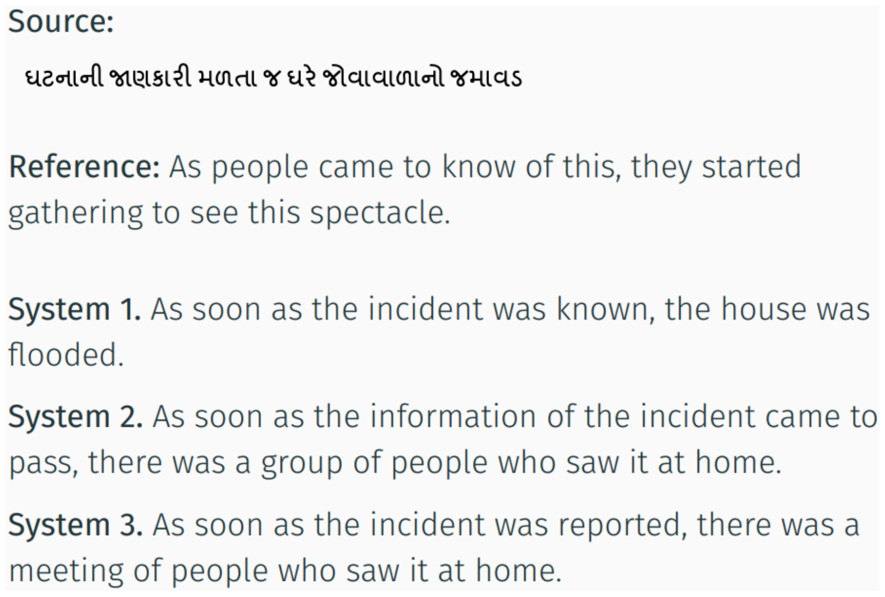
Source. Avauspelin voitto on aina tärkeä.

Reference. It is always important to win the opening match.

System 1. Victory for the game is always important.

System 2. The victory of the opening game is always important.

System 3. Victory in the opening game is always important.



Source:

ઘટનાની જાણકારી મળતા જ ઘરે જોવાવાળાનો જમાવડ

Reference: As people came to know of this, they started gathering to see this spectacle.

System 1. As soon as the incident was known, the house was flooded.

System 2. As soon as the information of the incident came to pass, there was a group of people who saw it at home.

System 3. As soon as the incident was reported, there was a meeting of people who saw it at home.

– moreover, **human evaluators** will often disagree about the **fitness** of a translation:

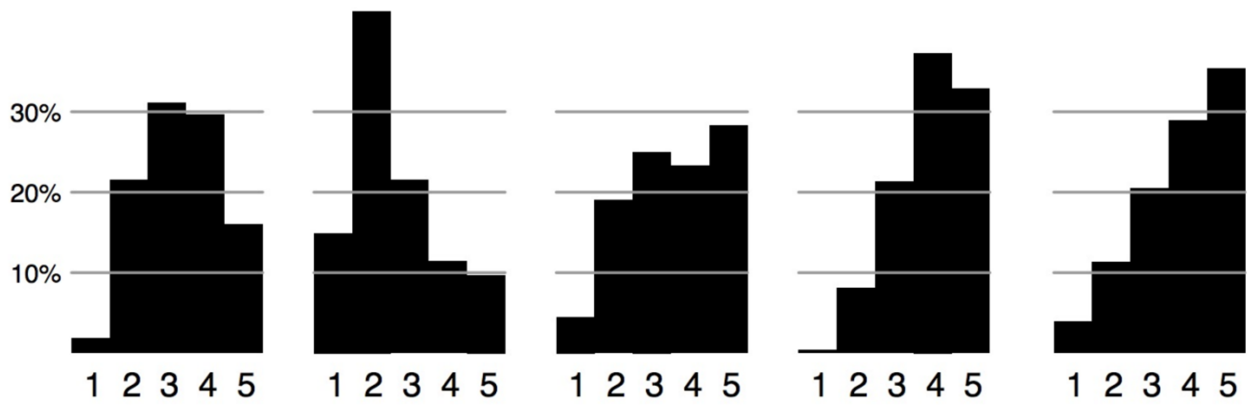


Figure 6: Scores assigned by a group of evaluators. Each histogram corresponds to a different translation-model. We can see that for all the models, some evaluators believe it has a poor performance, whilst others think it performs really well.

- **How can agreement between evaluators be measured?**

- we can use **Cohen's kappa**:

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

  where:
    - $P(A)$: proportion of times evaluators agree
    - $P(E)$: proportion of times evaluators would agree by chance
- the higher the κ, the more agreement there is
- in MT, $P(E)$ can be computed, for example, as the proportion of times in which 2 evaluators which assign a **random** score agree. If we have a score from 1 to 5, we'd have that:

$$P(E) = \frac{1}{5} \times \frac{1}{5} = \frac{1}{25}$$

- **Empirically, how high is agreement in translation?**
  - agreement on **fluency** and **adequacy** tends to be **low** but **positive**
  - agreement on rating (which translation is better?) is slightly higher
  - **adequacy** and **fluency** are abstract and difficult to measure
- **What is the 100-point Likert scale?**
  - a scale, from 1 to 100, to rate the quality of a translation, created during the **Workshop/Conference on Machine Translation**
  - more fine-grained, so allows better statistical analyses
  - for instance, if an annotator tends to rate everything highly, we can **normalise** values to be centered at 0, and remove outliers
  - tends to score higher **intra-annotator agreemenet** and **inter-annotator agreemenet**

2.4 Automatic Evaluation

- **Why is automatic evaluation necessary?**
  - human evaluation is **expensive** and **time-consuming**
  - if we want to evaluate **incremental** changes to our system, using **human evaluation** is unfeasible
  - moreover, in systems like **Google Translate**, we need to evaluate translations in more than 100 languages
- **What is the key difficulty in implementing automatic machine translation evaluators?**
  - we need to create a system which, given **reference translations** can rate the **generated translation**
  - doing this in a systematic, albeit "human-like" manner is complex: we can easily process different word orders, word choices, etc ... when determining whether a translation is good

2.4.1 Naive Approach

- **What is precision?**
  - the proportion of words in the **generated translation**, which are also part of the **reference translation**

- more generally:

$$precision = \frac{TP}{TP + FP}$$

  where TP is the number of **true positives** (words which are in both generated and reference translations) and FP is the number of **false positives** (words in generated translation which aren't in reference translation)

- **What is recall?**
  - the proportion of words in the **reference translation** which are also part of the **generated translation**
  - more generally:

$$recall = \frac{TP}{TP + FN}$$

    where FN is the number of **false negatives** (words in reference translation which aren't in the generated translation)

- **What is the F_1 score?**
  - the **harmonic mean** between **precision** and **recall**:

$$F_1 = \frac{2 \times precision \times recall}{precision + recall}$$

- **Why is precision alone a bad metric for evaluating machine translation?**
  - completely nonsensical translations can attain **perfect** precision:

    Reference: *The* cat is on *the* mat
    Candidate: *the the the the the the the the*

  - this gives perfect precision, since every word in the **candidate** ("the") is part of the reference

- **Why is a naive F_1 score a bad metric for evaluating machine translation?**
  - a **high** F_1 score can only be attained with **both** a high precision and a high recall
  - nonetheless, it can still be fooled, for instance, by just **permuting** the words in the reference translation:

    Reference: *The cat is on the mat*
    Candidate: *Cat the on mat is the*

  - this has a **precision** of 1 and a recall of 1, so $F_1 = 1$
  - other translations, which make a lot more syntactic sense won't be as good:

    Reference: The *cat is on the mat*
    Candidate: There *is* a *cat on the mat*

    which obtains:

$$precision = \frac{5}{7} \approx 0.71 \quad recall = \frac{5}{6} \approx 0.83 \quad F_1 \approx 0.77$$

  - in other words, we are not accounting for **word order**

2.4.2 The BLEU Score

- **What is the BLEU score?**
  - an **automatic** machine translation evaluator
  - considers:
    * many different (possible) references
    * word choice in candidate, compared to references
    * word ordering in candidate, compared to references
    * length difference between candidate and references
  - computed via:

$$BLEU = BP \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

  where:
    * N is the maximum number of **n-grams** to consider, when comparing the candidate with the references
    * BP is a **brevity penalty**, which penalises output sentences which are shorter than reference sentences:

$$BP = \begin{cases} 1, & c > r \\ e^{1-\frac{r}{c}}, & c \leq r \end{cases}$$

      where c is the length of the **candidate** translation, whilst r is the **effective reference corpus length** (the length of the reference translations that best match the length of the candidate translation being evaluated)
    * $w_n > 0$ is some positive weighting, such that:

$$\sum_{n=1}^{n} w_n = 1$$

    * $p_n \geq 0$ is the **modified n-gram precision**
- **What is the modified unigram precision?**
  - we saw above that **candidate translations** which just repeated a word from the **reference translations** would get **perfect precision**:

    Reference: *The* cat is on *the* mat
    Candidate: *the the the the the the the the*

  - the **modified unigram precision** resolved this, by **clipping** the number of times a **candidate word** can be used in counts when determining **precision**
  - for instance, in the example above, "the" appears only twice in the reference, so "the" will be counted **at most** twice when computing precision, for **any** candidate sentence. As a result, the **modified unigram precision** for this candidate is:

$$p_1 = \frac{2}{7}$$

  - formally, we define the **clipped count** of a word as:

$$Count_{clip}(w) = \min(Candidate\_Count(w), Max\_Ref\_Count(w))$$

    where $Max\_Ref\_Count(w)$ is the maximum number of times w appears in **any** of the reference translations (if we have "There is a cat on the mat" as another reference, we'd still have $Max\_Ref\_Count(the) = 2$, since "the" appears twice in the original reference)

- the **modified unigram precision** for a single word w is computed as:

$$p_1(w) = \frac{Count_{clip}(w)}{Candidate\_Count(w)}$$

- the **modified unigram precision** for a **candidate** sentence is:

$$p_1 = \frac{\sum_{w \in Candidate} Count_{clip}(w)}{\sum_{w \in Candidate} Candidate\_Count(w)}$$

- **How is modified unigram precision adapted for n-grams?**
  - we can defined the **modified n-gram precision** for a **candidate** via:

$$p_n = \frac{\sum_{\text{n-gram } \in Candidate} Count_{clip}(\text{n-gram})}{\sum_{\text{n-gram } \in Candidate} Candidate\_Count(\text{n-gram})}$$

  - $Candidate\_Count$ and $Count_{clip}$ work in the exact same way, but now for general **n-grams**

Consider the following:

Reference 1: The cat is on the mat.
Reference 2: There is a cat on the mat.
Candidate: The cat the cat on the mat.

*If we want to compute p_2, we need to consider **counts** and **clipped counts** of each **bigram** in the candidate:*

Bigram	$\boldsymbol{Count_{clip}}$	$\boldsymbol{Candidate\_Count}$
the cat	1	2
cat the	0	1
cat on	1	1
on the	1	1
the mat	1	1

For instance, "the cat" only appears once in each of the references, but twice in the candidate. Similarly, "cat the" never appears in any of the references, but appears once in the candidate.
With these counts, we can then compute:

$$p_2 = \frac{1+0+1+1+1}{2+1+1+1+1} = \frac{4}{6}$$

- **What is the purpose of the brevity penalty?**
  - shorter sentences are likely to have **high precision**

  - for instance, if a **candidate translation** is just "of the", it will obtain a 1 for **modified precision** so long as any of the references contain "of the" aswell
  - the **brevity penalty** penalises **short** translations, to ensure that a **candidate** is as close to the **references** as possible

- **Why is there no penalty for sentences which are longer than the references?**
  - **long candidate translations** are already inherently penalised by **precision** itself
  - the longer a sentence, the greater the **denominator** of the modified precision

- **How well does BLEU correlate with human judgement in MT?**
  - at the **system level** (i.e when evaluating the quality of a batch of translations together), **BLEU** seems to correlate **moderately** with **human evaluation**
  - however, at the **text level** (i.e judging the translation of each sentence individually), this correlation is **low**

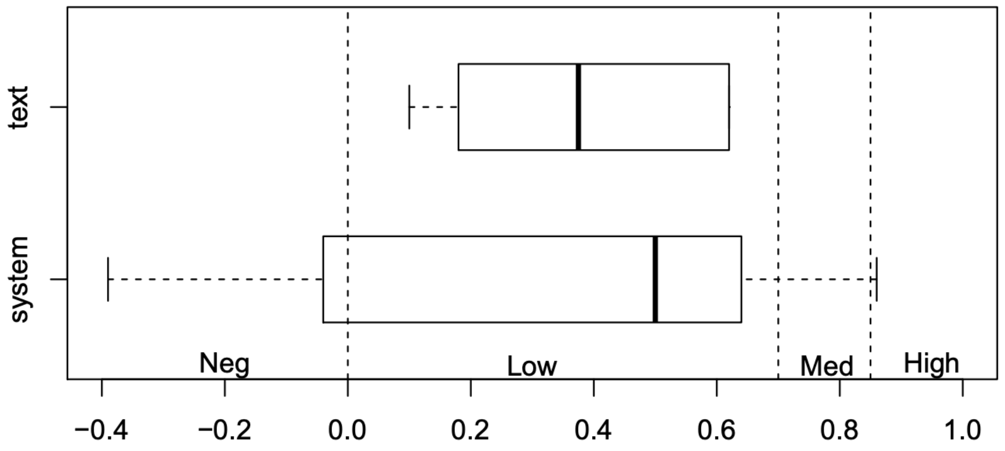
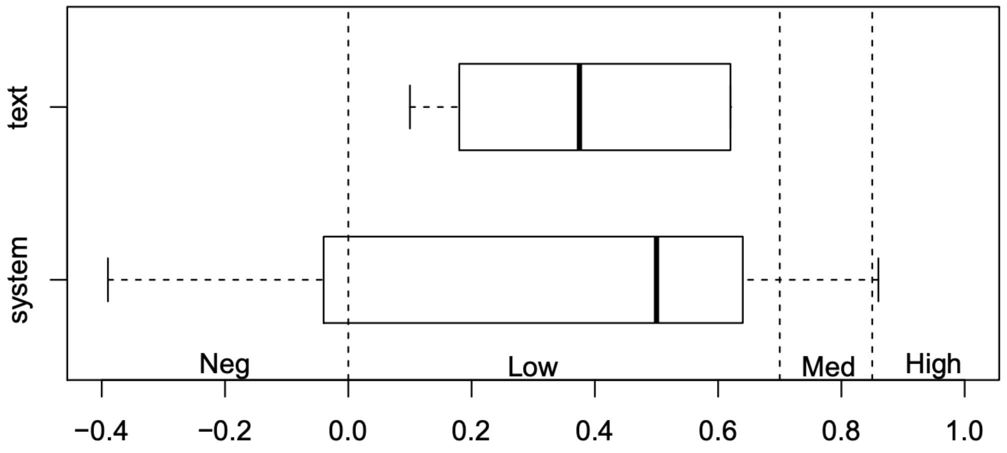


Figure 7: Correlation between **BLEU** evaluation and **human** evaluation. Figure from A Structured Review of the Validity of BLEU, by Ehud Reiter

- **How well does BLEU correlate with human judgement of NLG?**
  - for **natural language generation**, the **correlation** is **low**, both at **system** and **text** levels:



- **Why is BLEU just a crude measure?**
    1. **Word Importance**: **BLEU** will treat determiners and puncutation the same as names and content words, when the latter should be more important for a good translation
    2. **Surface-Level Features**: **BLEU** focuses on **lexical** and **syntactic** elements of text, which aren't the most important factors for **human-level** translation (i.e synonyms, different syntactic structures, etc...). As such, it might not be a good proxy for **adequacy** and **fluency**.
    3. **Bad References**: **BLEU** is biased towards reference translations, so it will assign higher scores to **candidates** which are most similar to these (even if the references might be written in an awkward manner)
    4. **Interpretability**: **BLEU** scores aren't too **comparable**, since **BLEU** scores don't have an **intrinsic meaning**; the evaluation for an English to French translation won't necessarily be comparable to those of an English to Chewa (a language in Zambia)
- **Despite these pitfalls, why was BLEU so widely used?**
    - it is a **simple** metric, which is easy to compute

2.4.3 Model-Based Evaluation: COMET

- **Why is the BLEU score no longer as useful?**
    - **BLEU** came out in 2002, when **machine tranlsation** systems were rather simple
    - nowadays, MT systems are so good that **BLEU** can no longer distinguish between different systems
    - we need a more **sensitive** measure
- **What is the COMET evaluation metric?**
    - **COMET** (**C**rosslingual **O**ptimized **M**etric for **E**valuation of **T**ranslation) is a **trained metric**: we learn a model which scores translations
    - use **transformers** to **pre-train** a **cross-lingual** language model
    - we then **fine-tune** the model using **human evaluations** (we have 16 years of such data from WMT (Conference on Machine Translation, previously known as Workshop on Machine Translation))

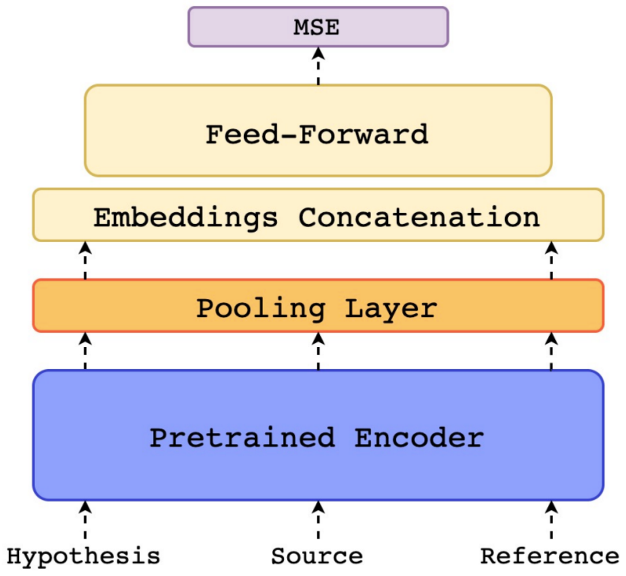


Figure 8: The **fine-tuning** process for COMET. As inputs it takes a **hypothesis** (the **candidate translation**), a **source** (the original text which is translated) and a **reference** (the **reference translation**). The **fine-tuning** objective is then to predict the **human evaluation** score (i.e a number from 1-100).

- **How well does COMET correlate with human understanding?**
  - **COMET** has obtained **state of the art** results for **correlation** with human judgement

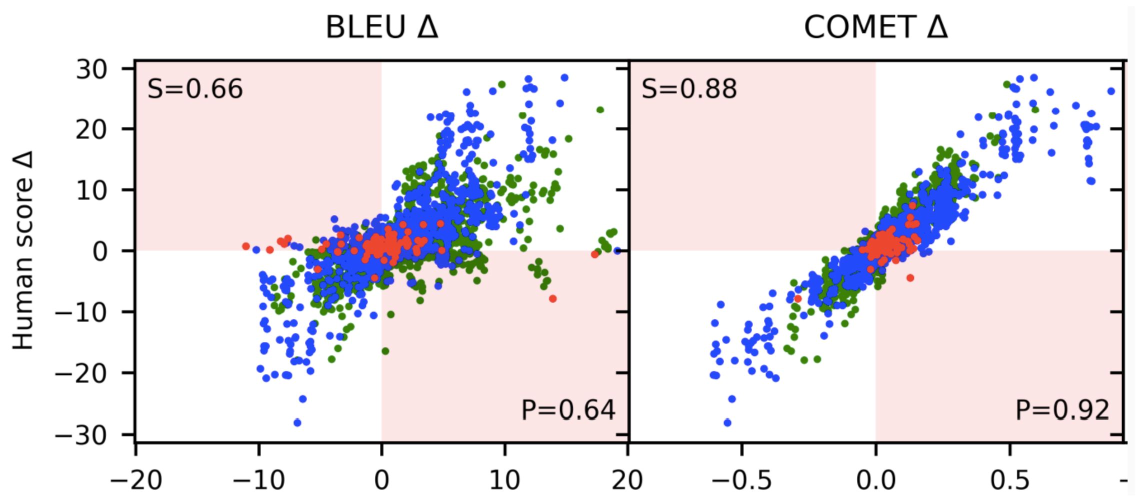


Figure 9: Comparison between the correlation of BLEU and COMET with human judgement. Each data point corresponds to a **pair** of **machine translation systems**. The x-axis corresponds to the **difference** in the scores assigned by **BLEU/COMET** for each system pair. The y-axis corresponds to the **difference** in the scores assigned by **human evaluators** for each system pair. A perfect diagonal line would correspond to perfect agreement between human evaluation and automatic evaluators. We can see that the COMET is a lot more correlated with humans, whilst BLEU seems to contradict human judgement for a lot of systems (represented by points which are in the red rectangles).

- **Do COMET scores still have problems?**

- the main issue with **COMET** is that scores aren't **bounded**
- in particular, they are much less **interpretable** than **BLEU** scores
- similar to **BLEU**, they are also not **comparable**

Takeaway Points:

- ***MT*** *(and* ***NLG****) are both* ***important*** *and* ***difficult***
- *Good MT systems are* ***adequate*** *and* ***fluent***
- ***BLEU*** *provides automatic evaluation, by considering n-gram overlap between* ***candidates*** *and* ***references*** *- however, this has many problems*
- ***Trained metrics****, like* ***COMET****, correlate better with humans skeptical of claims of human-level accuracy in MT - no metric is perfect*