SMT

Question 1:

Why is more data valuable for figuring out a language?

To improve the quality of translations, we need more data and of the right kind. SMT learns from two kinds of data:

- Source documents and their human translations.
- Target language collection.

More data means:

- Can learn more sentence alignments. (Word alignment + translation probabilities = translation model).
- Can learn more words that translate into other words
- Can get better understanding of what target sentences should look like.

If the corpora we have is small, of poor quality and not representative – then our statistical model will be poor. If the data **is** representative, we have more chance of identifying idioms/colloquial language.

The more data we have, the longer phrases we can learn, the longer phrases we can learn the more context we can take into account, the more context we can take into account the more non-compositional phrases or idioms we can handle – resulting in a better model.

Translate sentences by hand (similar to Q1 2016) describe how you did it, alignments, unknown words.

-Similar to how SMT works, look through data bit by bit, reduce search space, and phrase alignments.
-Create collated statistics table showing count of each word/phrase occurrence.

Question 2:

Why MT is important in this day & age? Example use cases.

MT is important in this day and age as it is helping companies cope with the explosion of data, there now more multilingual content than ever before – resulting in growing demand for rapid, instant communication.

Today's MT engines can be customized to fit a customer's style, terminology and industry sector. The principal use case being localization for publication.

Use cases: Adobe & ProMT, Ford & Sytran, Dell & Safaba

Traditional Use Cases	Emerging Use Cases
Raw MT: Internal communications (assimilation), emails, FAQs, gistings, product listings.	Raw MT: Huge volumes of UGC (User Generated Content) Hotel / product reviews, online chat, social posts. User demand: Counting hits on translated websites.
Light Post Editing: Online help, Support Documentation, Knowledge Forums	Real time translation: Multilingual chat, **Only** MT can facilitate that. **Only** MT can provide confidentiality **Only** MT can rapidly facilitate to fit as customer's style and requirements in a short time.
Full Post Editing: External Communication, Sensitive Documents (Security / Health concern), Client facing documents.	

Why clients would want customized translations?

- Improve productivity
- Translate content previously not feasible due to time and cost restraints.
- Reduce time to market and translation costs.
- Client specific assets (style, data, terminology, industry sector)

Why do some people believe MT is not useful?

Many translators find MT as a useful tool in their armory on a daily basis but that's all MT is, a useful too. There's no threats to translators' jobs, rather making translators more productive and efficient. The time is ripe for MT developers and translators to work together and further advance translation.

What is the role of the human in the MT pipeline?

The role of the human is typically involved in the post-editing scenarios: light post-editing and full post-editing. The human reviews and post-edits to make the MT output an understandable reflection of the source with added stylistic niceties within full post-editing.

We can learn a lot as engineers from human translators:

- We can make use of experience and mental outlook.
- A human can concentrate on context and solve ambiguity.

The degree of human involvement warranted depends on the purpose, value and shelf life of the content.

Why is MT hard?

Human languages are complex with varying underlying structures:

- Many ways of saying the same thing.
- Meaning depends on context.
- One word / sentence may mean many things.
- Word order.
- Cultural differences – idioms / metaphors.

Question 3:

Given two types of linguistic information: syntactic and morphology. Give examples of where these types of information could be incorporated within pre-processing, decoding (translation), post-processing to improve the MT system output.

Linguistic knowledge:

Linguistic knowledge can improve MT output but can increase complexity.

Why bother with linguistic knowledge?

- Data driven SMT works well but there is room for improvement.
- Data driven SMT relies on large corpus, for minority languages this can be a problem.

Using linguistic knowledge we can achieve:

- Get more global view of fluency than an n-gram model.
- Long distance dependencies.
- Word compounding.

How can linguistic knowledge be encoded into an SMT system?:

- Before translation (pre-processing)
- During translation (decoding)
- After translation (post-processing)

Pre-processing:

- Word (or morph) segmentation: The notion of a word varies from language to language. In Arabic (lecturer is Arabic so he will probably give English – Arabic), word segmentation involves breaking words into constituent prefix, stem, suffix. This can be a problem for MT.

  Potential Solution: Automatic word segmentation before translation.

- Word lemmatization: Languages with rich inflectional system can have several variants of the same root form or lemma. e.g war, wars in Latin bellum, belli, bello – the um, I, o.

  Potential Solution: Before translation, replace full form with lemma.

- Syntactic Re-ordering: Harder to translate between languages with different word order: Subject Verb Object (SVO), SOV, VSO.

  Potential Solution: Transform the source sentence so it's word order more closely resembles the word order of target language.

During Translation:

- Log Linear Models: Log linear model of translation allows several sources of information to be applied simultaneously during the translation process. Some of this information can be linguistic knowledge, instead of calculating p(ocras | hungry) we can calculate p(noun | adjective)

- Tree based Models: replace phrase based translation model with a different type model. Tree based model based on syntax tree. Allows in theory to better capture structural similarities/divergences between languages.

- Syntactic language Models: Instead of computing the probability of a sentence by multiplying n-gram probabilities, the probability is computed by multiplying the rules that are used to build the syntax trees.

Post-processing:

- Rules

- Re-Ranking: SMT systems return a ranked list of candidate translations. Linguistic information of all kinds can be employed in re-ranking this list using machine learning techniques.

QUESTION 4:

GIVEN A SENTENCE, CALCULATE THE N-GRAMS AND THEN CALCULATE BLEU – Language Modelling Question
Language Modelling: A language model is the probabilistic distribution of all possible sentences. I.e. How likely it is that e within p(e) is an acceptable sentence. Measures fluency too. Translation Model measures adequacy.

Markov Assumption: Can approximate the i'th word by observing the shortened context history of the preceding n-1 words rather than the whole history (i-1 words)

Probability of an n-gram=
Bigram: $P(y|x)$ = count of x y / count of x
Trigram= $p(z|x\ y)$ = count of x y z / count of x y

Language model smoothing:
Smoothing in language model is the process by which unseen events are given a non-zero probability. This is achieved by some combination of:

- Count adjustment
- Interpolation
- Back-off

Add alpha smoothing:

c + alpha / n + v(alpha)

n = count of history
v = vocabulary size
c = count of n-gram
alpha = given in question

Q3(b): 2016

<s> Richard likes Ada </s>

<s> Richard hates Ada </s>

	Denis	likes	Ada	Richard	hates	java	</s>	Sum
<s>	1	0	2	0	0	0	0	3
Denis	0	1	0	0	0	0	0	1
likes	0	0	1	1	0	0	0	2
Ada	0	1	0	0	1	0	1	3
Richard	0	0	0	0	0	0	1	1
hates	0	0	0	0	0	1	0	1
java	0	0	0	0	0	0	1	0

	Denis	likes	Ada	Richard	hates	java	</s>	Sum
<s>	1+0.3	0+0.3	2+.3	0+.3	0+.3	0+.3	0+.3	3 + 2.1
Denis	0+.3	1+.3	0+.3	0+.3	0+.3	0+.3	0+.3	1 + 2.1
likes	0	0	1	1	0	0	0	2 + 2.1
Ada	0	1	0	0	1	0	1	3 + 2.1
Richard	0	0	0	0	0	0	1	1 + 2.1
hates	0	0	0	0	0	1	0	1 + 2.1
java	0	0	0	0	0	0	1	1 + 2.1

<s> Richard	Richard likes	likes Ada	Ada </s>
0 + 0.3 / 3 + 2.1	0 + 0.3 / 1 + 2.1	1 + 0.3 / 2 +2.1	1 + 0.3 / 3 + 2.1
<s> Richard	Richard hates	hates Ada	Ada </s>
0 + 0.3 / 3 + 2.1	0 + 0.3 / 1 + 2.1	0 + 0.3 / 1 + 2.1	1 + 0.3 / 3 + 2.1

Language model evaluation: Perplexity

The quality of a language model can be estimated using perplexity.

Perplexity measures the cross entropy between the empirical distribution (the distribution of things that actually appear) and the predicted distribution (what your model likes) and then divides by the number of words and exponentiates after throwing out unseen words.

Lower perplexity means language model m is better than language model n.

Minimizing perplexity = maximizing probability.

Bleu:

The closer a machine translation is to a human translation, the better it is. This is the central idea behind Bleu. Bleu is the computation of n-gram overlap between output and reference.

An artificially high score can be obtained by minimizing the length of the translation, to prevent this – the brevity penalty is used.

Brevity penalty: if output length is greater than reference length, brevity penalty = 1. If output length is shorter than reference length, brevity penalty = output length / reference length.
Example:

Reference = Salmons swim in the river
Candidate = Salmon swim in the river
Brevity penalty = output length / reference = 5/5 = 1
Number of n-grams (N) = 3
P1: unigram: number of correct tokens in output / output length = 4/5
P2: bigram:
Ref: (Salmons swim) (swim in) (in the) (the river)
Candidate: (Salmon swim) (swim in) (in the) (the river)
# of correct = 3
Probability = number of correct tokens in output / output length = ¾
P3: trigram:
Ref: (Salmons swim in) (swim in the) (in the river)
Candidate: (Salmon swim in) (swim in the) (in the river)
# of correct = 2
Probability = number of correct tokens in output / output length = 2/3
Bleu = 5/5 = 1
1 ($\sqrt[3]{}$ 4/5 x 3/4 x 2/3) = 0.73

Question 5 – Word Alignment, EM Algorithm


EM Algorithm

Question 6 – decoding - not doing

Question 7 – Neural Machine Translation

NMT is an approach to machine translation that uses a large neural network i.e modelling the entire MT process via one big artificial neural network.

It departs from phrase based MT in that it uses separately engineered subcomponents.

Unlike conventional MT systems, all parts of the NMT model are trained jointly (end-to-end) to maximize translation.

NMT systems work by encoding a source sentence into a vector using a Recurrent Neural Network (RNN) and then decoding a target sentence based on that vector, also using a RNN.

3 big wins for NMT:

- End to End training
- Better exploitation of word & phrase similarities
- Better exploitation of context.

Activation function:

An activation function defines the output of a given node given an input or a set of inputs.
The role of activation function is to make neural networks non-linear.
Why? If the data we wish to model is non-linear then we need to account for that in our model. An activation function then allows us to model a response variable (aka target variable, class label, or score) that varies non-linearly with its explanatory variables – meaning that it cannot be reproduced from a linear combination of the inputs (affine).

Activation functions are there to give neural nets the power they have, if you remove the activation function then your net has degenerated into a linear transformation, which is not strong enough to model any data.

Without activation functions you're just adding a lot more parameters to your model, making it slower to train and make inference from.

Derivation of sigmoid function over x: $o(x) = 1 / 1+ e^{x}$

$$o'(x) = o(x)(1 - o(x)) = 1 / (1 + e^{x})(1 + e^{-x})$$

Attention Mechanism:

The attention mechanism in neural nets are loosely based on human visual attention mechanism i.e. being able to focus on a certain region or a patch of image with "high resolution" while perceiving the surrounding image in "low resolution" and then adjusting the focal point over time.

What problem does attention solve in NMT? In NMT we map the meaning of a sentence into a vector representation and then generate a translation based on that vector. By not relying on things like n-gram counts and instead trying to capture higher level meaning of a text. NMT systems generalize to new sentences better.

The attention mechanism indicates:

- An implicit soft alignment between predicted target words against source-side words.
- The weights are jointly learned when training the NMT system with a feed forward Back Propagation algorithm. The sum of all weights is 1.

The Back Propagation (BP) algorithm:

The BP algorithm is the work horse of learning in NMT. It is also used to train a RNN.

The goal of BP is to optimize the weights so that the neural network can learn to quickly map arbitrary inputs to outputs.

The BP algorithm updates each of the weights in the network so that they cause the actual output to be closer to the target output – minimizing error for each output neuron and the network as a whole.

BP is a way of computing the network's weights by combining the chain rule, along with gradient descent.