Machine Translation: From Rules to Neural Networks

CSCI E-89B: Introduction to Natural Language Processing

Lecture 13

Lecture Information

Course: CSCI E-89B: Introduction to Natural Language Processing
Lecture: 13 – Machine Translation: Theory and Practice
Institution: Harvard Extension School
Topics: Machine Translation, Seq2Seq, Attention, Transformers, BLEU Score, Hybrid Models

Contents

1 Introduction to Machine Translation

Overview

This lecture provides a comprehensive exploration of **Machine Translation (MT)**—the task of automatically converting text from one language to another. We trace the evolution from rule-based systems through statistical methods to modern neural approaches, culminating in the transformer architecture that powers today's state-of-the-art translation systems.

1.1 What is Machine Translation?

Definition

Machine Translation (MT): The use of software to translate text or speech from one natural language to another, with the goal of producing output that is:

- Semantically accurate (preserves meaning)
- Grammatically correct (follows target language rules)
- Culturally appropriate (handles idioms and cultural references)
- Fluent and natural (reads like human-written text)

1.2 Why is Translation Difficult?

Translation is far more complex than word-for-word substitution. Consider the challenges:

1.2.1 1. Lexical Ambiguity

Example

The Word "Bank":

- "I deposited money at the bank." → Financial institution
- "I sat by the river bank." → Edge of water
- "Don't bank on it." → Rely/depend

Without context, the system cannot know which meaning to use!

1.2.2 2. Structural Differences

Different languages have different word orders:

- **English (SVO):** "The cat ate the fish."
- **Japanese/Korean (SOV):** "The cat the fish ate."
- **Arabic/Welsh (VSO):** "Ate the cat the fish."

Warning

The Reordering Problem:
To translate "I will go to school tomorrow" from English to Japanese, the system needs to:

1. Read the entire sentence
2. Understand the structure
3. Rearrange to: "I tomorrow school to will go"

This requires understanding the *entire* sentence before producing output!

1.2.3 3. Idiomatic Expressions

Example

Idioms Cannot Be Translated Literally:

English Idiom	Literal Translation	Correct Translation
"It's raining cats and dogs"	Animals falling	"It's raining heavily"
"Break a leg"	Fracture your bone	"Good luck"
"Shoot the breeze"	Fire at wind	"Chat casually"
"Hit the nail on the head"	Strike metal	"Exactly right"

1.2.4 4. Context and Coreference

Example

Pronoun Resolution:
"The trophy doesn't fit in the suitcase because it is too big."
What does "it" refer to?

- If "it" = trophy → The trophy is too large
- If "it" = suitcase → Wait, that doesn't make sense here

Compare: "The trophy doesn't fit in the suitcase because it is too small."

- Now "it" = suitcase (the suitcase is too small)

Resolving this requires world knowledge and reasoning!

Table 1: Evolution of Machine Translation Approaches

Era	Approach	Key Characteristics
1950s–1980s	Rule-Based (RBMT)	Hand-crafted rules, dictionaries, linguistic knowledge
1990s–2010s	Statistical (SMT)	Learn from parallel corpora, phrase-based, n-gram models
2014–Present	Neural (NMT)	End-to-end deep learning, Seq2Seq, Attention, Transformers

2 Historical Evolution of Machine Translation

2.1 Timeline Overview

2.2 Rule-Based Machine Translation (RBMT)

Definition

RBMT: Translation systems that use hand-crafted linguistic rules, bilingual dictionaries, and grammar specifications created by human experts.

2.2.1 How RBMT Works

1. **Analysis:** Parse the source sentence to understand its grammatical structure
2. **Transfer:** Apply rules to convert source structure to target structure
3. **Generation:** Produce the output sentence following target language grammar

Example

RBMT Rule Example:
English to Spanish rule for adjective placement:

```
IF: English has [Adjective] [Noun]
THEN: Spanish uses [Noun] [Adjective]

Input:  "the red car"
        [Det] [Adj] [Noun]
Output: "el coche rojo"
        [Det] [Noun] [Adj]
```

2.2.2 Advantages of RBMT

- **Precision:** Very accurate for well-defined domains (weather reports, legal documents)
- **Transparency:** Easy to understand why a translation was produced
- **No data required:** Works without large parallel corpora
- **Consistency:** Same input always produces same output

2.2.3 Disadvantages of RBMT

- **Labor-intensive:** Requires years of expert work to create rules
- **Incomplete coverage:** Cannot handle exceptions not anticipated by rule writers
- **Poor generalization:** Rules for one domain don't transfer to others
- **Unnatural output:** Often produces grammatically correct but stilted text

2.3 Statistical Machine Translation (SMT)

Definition

SMT: Translation systems that learn statistical patterns from large collections of parallel texts (texts and their human translations).

2.3.1 The Noisy Channel Model

SMT is based on Bayes' theorem. To find the best translation $\hat{e}$ of foreign sentence f:

$$\hat{e} = \arg\max_e P(e|f) = \arg\max_e P(f|e) \cdot P(e) \tag{1}$$

where:

- $P(f|e)$ = **Translation model**: How likely is f given e?
- $P(e)$ = **Language model**: How likely is e to be a valid sentence?

Example

SMT Intuition:
To translate French "le chat" to English:
Translation model says:

- P("le chat"|"the cat") = 0.8
- P("le chat"|"cat the") = 0.1

Language model says:

- P("the cat") = 0.01 (common phrase)
- P("cat the") = 0.0001 (ungrammatical)

Combined: "the cat" wins because it's both a good translation AND good English.

2.3.2 Phrase-Based SMT

The breakthrough came with **phrase-based models** that translate multi-word phrases as units:

- "in spite of" → "a pesar de" (as one unit)
- Better than word-by-word: "in" → "en", "spite" → "rencor", "of" → "de"

2.3.3 Advantages of SMT

- **Data-driven:** Learns from examples, no manual rules needed
- **Better coverage:** Can handle diverse vocabulary
- **Scalable:** More data = better translations

2.3.4 Disadvantages of SMT

- **Local context only:** Phrases are translated independently, losing global coherence
- **Reordering issues:** Hard to model long-distance word movement
- **Complex pipelines:** Many separate components to tune
- **Data hungry:** Needs millions of sentence pairs

2.4 Neural Machine Translation (NMT)

Definition

NMT: Translation systems that use deep neural networks to learn a direct mapping from source to target language in an end-to-end fashion.

2.4.1 Key Advantages of NMT

- **End-to-end learning:** Single model learns everything jointly
- **Continuous representations:** Words/phrases are vectors, enabling generalization
- **Global context:** Can consider entire sentence when translating each word
- **Fluent output:** Produces more natural-sounding translations

3 Sequence-to-Sequence Models

3.1 Architecture Overview

The Seq2Seq model, introduced in 2014, consists of two main components:

Definition

Seq2Seq Architecture:

1. **Encoder:** Reads the source sentence and compresses it into a fixed-length vector (context vector)
2. **Decoder:** Takes the context vector and generates the target sentence word by word

3.1.1 The Encoder

The encoder processes the input sequence one token at a time:

$$h_t = \text{RNN}(h_{t-1}, x_t) \tag{2}$$

$$c = h_T \quad \text{(final hidden state = context vector)} \tag{3}$$

where:

- x_t = input token at time t
- h_t = hidden state at time t
- c = context vector (summary of entire input)

3.1.2 The Decoder

The decoder generates output tokens one at a time:

$$s_t = \text{RNN}(s_{t-1}, y_{t-1}, c) \tag{4}$$

$$P(y_t|y_{<t}, x) = \text{softmax}(W_o \cdot s_t) \tag{5}$$

where:

- s_t = decoder hidden state
- y_{t-1} = previous output token
- c = context vector from encoder

3.2 The Bottleneck Problem

Warning

Critical Limitation:
The entire source sentence must be compressed into a single fixed-length vector!
Analogy: Imagine reading a 500-page novel and summarizing it in a single sentence, then asking someone to recreate the entire novel from just that sentence.
Consequences:

- Information loss, especially for long sentences
- Performance degrades significantly as sentence length increases
- No way to "go back" and look at specific parts of the input

Example

Empirical Evidence:
Early Seq2Seq models showed BLEU scores dropping dramatically for sentences longer

than 20 words. The context vector simply couldn't capture all the necessary information.

4 Attention Mechanism in Translation

4.1 The Core Idea

Important

Key Insight: Instead of compressing everything into one vector, let the decoder "look back" at the input at each step and focus on the most relevant parts!

Definition

Attention Mechanism: A method that allows the decoder to access all encoder hidden states and compute a weighted average based on relevance to the current decoding step.

4.2 How Attention Works for Translation

4.2.1 Step 1: Compute All Encoder Hidden States

Unlike basic Seq2Seq, we keep ALL hidden states:

$$H = [h_1, h_2, \ldots, h_T] \tag{6}$$

4.2.2 Step 2: Compute Attention Scores

For each decoder step t, compute how relevant each encoder state is:

$$e_{ti} = \text{score}(s_{t-1}, h_i) \tag{7}$$

Common scoring functions:

- **Dot product:** $e_{ti} = s_{t-1}^T h_i$
- **Bilinear:** $e_{ti} = s_{t-1}^T W h_i$
- **Additive:** $e_{ti} = v^T \tanh(W_s s_{t-1} + W_h h_i)$

4.2.3 Step 3: Convert to Probabilities

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{j=1}^{T} \exp(e_{tj})} \tag{8}$$

4.2.4 Step 4: Compute Context Vector

$$c_t = \sum_{i=1}^{T} \alpha_{ti} h_i \tag{9}$$

4.2.5 Step 5: Generate Output

$$s_t = \text{RNN}(s_{t-1}, [y_{t-1}; c_t]) \tag{10}$$

$$P(y_t) = \text{softmax}(W_o[s_t; c_t]) \tag{11}$$

4.3 Attention Visualization

Example

English to French Translation:
Source: "The agreement on the European Economic Area was signed in August 1992"
When generating the French word "zone" (area), the attention mechanism assigns high weight to the English word "Area" and lower weights to other words.
This creates an **alignment matrix** showing which source words are most relevant for each target word.

	The	agreement	...	Area	1992
L'	**0.9**	0.05	...	0.01	0.01
accord	0.1	**0.8**	...	0.02	0.01
...	...	...	...	...	...
zone	0.01	0.05	...	**0.85**	0.02
1992	0.01	0.01	...	0.01	**0.95**

4.4 Benefits of Attention

Summary

Why Attention Revolutionized Translation:

1. **No bottleneck:** Each decoding step has access to all encoder states
2. **Long sentences:** Performance doesn't degrade with length
3. **Alignment learning:** Model learns word correspondences automatically
4. **Interpretability:** Can visualize what the model focuses on
5. **Variable-length handling:** Works for any sequence length

5 Transformers for Translation

5.1 From RNN+Attention to Pure Attention

Important

The key insight of Transformers: If attention is so powerful, **do we even need the RNN?**
Answer: **No!** The 2017 paper "Attention Is All You Need" showed that a model using *only* attention mechanisms can outperform RNN-based models.

5.2 Advantages Over RNN-Based Models

Table 2: Transformer Advantages for Translation

Feature	RNN+Attention	Transformer
Parallel training	No	Yes
Long-range dependencies	Difficult	Easy
Training speed	Slow	Fast
Gradient flow	Problematic	Stable

5.3 Self-Attention in Translation

In transformers, **self-attention** allows each word to attend to all other words in the same sentence:

Example

Self-Attention Example:
"The animal didn't cross the street because it was too tired."
Self-attention helps the model learn that "it" refers to "animal" (not "street") by allowing direct connections between these words.

5.4 The Encoder-Decoder Structure

Definition

Transformer for Translation:

- **Encoder:** 6 identical layers of (Self-Attention + Feed-Forward)
- **Decoder:** 6 identical layers of (Masked Self-Attention + Encoder-Decoder Attention + Feed-Forward)

5.4.1 Encoder Self-Attention

Every position in the source sentence can attend to every other position:

- Word "bank" can see "deposit" later in the sentence
- This helps disambiguate word meanings

5.4.2 Masked Self-Attention in Decoder

During generation, we can only attend to previously generated words:

- Prevents "cheating" by looking at future words
- Implemented by masking future positions with $-\infty$ before softmax

5.4.3 Encoder-Decoder Attention

Each decoder layer attends to the encoder output:

- Query: current decoder state
- Keys and Values: encoder outputs
- This is analogous to traditional attention over the source

5.5 Multi-Head Attention for Translation

Using multiple attention heads allows the model to capture different types of relationships:

Example

What Different Heads Learn:

- Head 1: Syntactic relationships (subject-verb agreement)
- Head 2: Semantic relationships (synonyms, related concepts)
- Head 3: Position-based patterns (nearby words)
- Head 4: Long-range dependencies (coreference)

5.6 Positional Encoding for Word Order

Since transformers process all positions simultaneously, word order must be explicitly encoded:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{12}$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{13}$$

Info

Why Sinusoidal Functions?

- Create unique encoding for each position
- Allow model to learn relative positions
- Generalize to longer sequences than seen during training

6 Evaluating Machine Translation: BLEU Score

6.1 The Need for Automatic Evaluation

Human evaluation is the gold standard but:

- Expensive and time-consuming
- Not scalable for rapid development
- Subjective and inconsistent

Definition

BLEU (Bilingual Evaluation Understudy): An automatic metric that measures how similar machine translation output is to human reference translations.

6.2 How BLEU Works

6.2.1 Step 1: N-gram Precision

Count how many n-grams in the machine translation appear in the reference:

$$p_n = \frac{\sum_{\text{n-gram} \in \text{candidate}} \text{Count}_{\text{clip}}(\text{n-gram})}{\sum_{\text{n-gram} \in \text{candidate}} \text{Count}(\text{n-gram})} \tag{14}$$

where $\text{Count}_{\text{clip}}$ prevents counting the same reference n-gram multiple times.

Example

Calculating Unigram Precision:
Reference: "The cat sat on the mat"
Candidate: "The the the the"
Without clipping: precision = 4/4 = 100% (wrong!)
With clipping: "the" appears 2 times in reference
Clipped count = min(4, 2) = 2
Precision = 2/4 = 50%

6.2.2 Step 2: Brevity Penalty

Prevent gaming the system by outputting very short translations:

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{1-r/c} & \text{if } c \leq r \end{cases} \tag{15}$$

where c = candidate length, r = reference length.

6.2.3 Step 3: Final BLEU Score

$$\boxed{\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)} \tag{16}$$

Typically $N = 4$ and $w_n = 1/4$ (uniform weights).

Example

BLEU Score Interpretation:

BLEU Score	Quality Level
< 10	Almost useless
$10 - 19$	Hard to understand
$20 - 29$	Gist is clear
$30 - 39$	Understandable
$40 - 49$	Good quality
$50 - 59$	Very good quality
$60+$	Near human quality

Note: Human translators typically score 60-80 on BLEU against other humans.

6.3 Limitations of BLEU

Warning

BLEU Has Significant Limitations:

1. **Synonym blindness:** "beautiful" vs "pretty" are both correct but BLEU penalizes
2. **Word order flexibility:** Some languages allow multiple valid orderings
3. **Meaning ignorance:** Can't detect semantic errors
4. **Single reference bias:** One reference may not cover all valid translations

Example

BLEU Failure Case:
Reference: "The quick brown fox jumps over the lazy dog."
Candidate A: "A fast brown fox leaps over a lazy dog." (Semantically perfect)
BLEU: **Low** (different words)
Candidate B: "The quick brown jumps fox over lazy the dog." (Nonsensical)
BLEU: **Higher** (more matching n-grams!)

6.4 Alternative Metrics

- **METEOR:** Considers synonyms and stemming
- **TER (Translation Edit Rate):** Number of edits needed

- **COMET:** Neural metric trained on human judgments
- **BERTScore:** Uses BERT embeddings for semantic similarity

7 Hybrid Translation Systems

7.1 Why Combine Approaches?

Definition

Hybrid MT: Systems that combine multiple translation approaches (neural, statistical, rule-based) to leverage their respective strengths.

7.1.1 Weaknesses of Pure NMT

- **Rare words:** May hallucinate translations for uncommon terms
- **Domain-specific terminology:** Medical/legal terms need exact translations
- **Consistency:** May translate the same term differently
- **"Hallucination":** Sometimes generates fluent but completely wrong content

7.1.2 Strengths of Rule-Based Components

- **Precision:** Exact translation of technical terms
- **Consistency:** Same term always translates the same way
- **Controllability:** Can enforce specific rules

7.2 Hybrid Architecture Examples

1. **Pre-processing:** Use rules to identify and protect special terms before NMT
2. **Post-processing:** Use rules to fix grammar or terminology after NMT
3. **Ensemble:** Combine outputs from multiple systems
4. **Terminology injection:** Force specific translations into NMT output

Example

Medical Translation Pipeline:

1. **Step 1 (Rule-based):** Identify medical terms ("myocardial infarction")
2. **Step 2 (NMT):** Translate general text fluently
3. **Step 3 (Rule-based):** Replace medical terms with verified translations
4. **Step 4 (Rule-based):** Check grammar and formatting

Result: Fluent translation with guaranteed accurate medical terminology.

7.3 Real-World Hybrid Systems

- **SYSTRAN:** Pioneer in hybrid MT, combines neural with rules
- **Microsoft Translator:** Uses neural models with terminology databases
- **DeepL:** Primarily neural but with extensive post-processing
- **Google Translate:** Neural with fallbacks for rare languages

8 Practical Implementation

8.1 Building a Simple Seq2Seq Model

Listing 1: Basic Seq2Seq Encoder-Decoder

```python
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# Encoder
class Encoder(keras.Model):
    def __init__(self, vocab_size, embedding_dim, enc_units):
        super().__init__()
        self.embedding = layers.Embedding(vocab_size, embedding_dim)
        self.lstm = layers.LSTM(enc_units, return_state=True)

    def call(self, x):
        x = self.embedding(x)
        output, state_h, state_c = self.lstm(x)
        return state_h, state_c

# Decoder with Attention
class Decoder(keras.Model):
    def __init__(self, vocab_size, embedding_dim, dec_units):
        super().__init__()
        self.embedding = layers.Embedding(vocab_size, embedding_dim)
        self.lstm = layers.LSTM(dec_units, return_sequences=True,
                                return_state=True)
        self.fc = layers.Dense(vocab_size)
        self.attention = layers.Attention()

    def call(self, x, encoder_output, states):
        x = self.embedding(x)
        context = self.attention([x, encoder_output])
        x = tf.concat([context, x], axis=-1)
        output, state_h, state_c = self.lstm(x, initial_state=states)
        output = self.fc(output)
        return output, state_h, state_c
```

8.2 Using Pre-trained Translation Models

Listing 2: Using Hugging Face Transformers

```python
from transformers import MarianMTModel, MarianTokenizer

# Load pre-trained English to French model
model_name = 'Helsinki-NLP/opus-mt-en-fr'
tokenizer = MarianTokenizer.from_pretrained(model_name)
model = MarianMTModel.from_pretrained(model_name)

# Translate
text = "Machine translation has come a long way."
inputs = tokenizer(text, return_tensors="pt", padding=True)
translated = model.generate(**inputs)
output = tokenizer.decode(translated[0], skip_special_tokens=True)
print(output)
# "La traduction automatique a parcouru un long chemin."
```

8.3 Computing BLEU Score

Listing 3: BLEU Score Calculation

```python
from nltk.translate.bleu_score import sentence_bleu, corpus_bleu

# Single sentence BLEU
reference = [['the', 'cat', 'sat', 'on', 'the', 'mat']]
candidate = ['the', 'cat', 'is', 'on', 'the', 'mat']
score = sentence_bleu(reference, candidate)
print(f"Sentence BLEU: {score:.4f}")

# Corpus BLEU (multiple sentences)
references = [[['the', 'quick', 'brown', 'fox']],
              [['jumped', 'over', 'the', 'dog']]]
candidates = [['the', 'fast', 'brown', 'fox'],
              ['jumped', 'over', 'a', 'dog']]
corpus_score = corpus_bleu(references, candidates)
print(f"Corpus BLEU: {corpus_score:.4f}")
```

9 Current State and Future Directions

9.1 State-of-the-Art Systems

Info

Leading Translation Systems (2024):

- **Google Translate:** 100+ languages, neural-based
- **DeepL:** Known for quality in European languages

- **Microsoft Translator:** Integrated into Office products
- **GPT-4/Claude:** Large language models with translation capabilities

9.2 Remaining Challenges

1. **Low-resource languages:** Many languages lack training data
2. **Document-level translation:** Maintaining coherence across paragraphs
3. **Multimodal translation:** Combining text, speech, and images
4. **Real-time translation:** Simultaneous interpretation
5. **Cultural adaptation:** Beyond literal translation

9.3 Emerging Trends

- **Multilingual models:** Single model for many language pairs
- **Zero-shot translation:** Translating between languages not seen together in training
- **Large language models:** GPT-4 and similar models as general-purpose translators
- **Human-in-the-loop:** Interactive translation with human feedback

10 One-Page Summary

Summary

Key Concepts from Lecture 13:

1. Machine Translation Evolution:

- **RBMT** (1950s-80s): Hand-crafted rules, precise but limited
- **SMT** (1990s-2010s): Statistical patterns from parallel data
- **NMT** (2014-present): End-to-end neural networks

2. Translation Challenges:

- Lexical ambiguity (bank = financial vs. river)
- Structural differences (SVO vs. SOV word order)
- Idioms (literal translation fails)
- Context and coreference resolution

3. Seq2Seq Models:

- Encoder → Context Vector → Decoder
- Problem: Bottleneck—everything compressed into one vector

- Solution: Attention mechanism

4. **Attention:**
   - Dynamic weighted sum of encoder states
   - Context vector: $c_t = \sum_i \alpha_{ti} h_i$
   - Enables handling of long sentences
5. **Transformers:**
   - "Attention Is All You Need" (2017)
   - Self-attention for parallel processing
   - Multi-head attention for diverse relationships
   - Foundation for BERT, GPT, modern NMT
6. **BLEU Score:**

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

   - Measures n-gram overlap with reference
   - Brevity penalty prevents short outputs
   - Limitation: ignores synonyms and meaning
7. **Hybrid Systems:**
   - Combine NMT fluency with rule-based precision
   - Important for domain-specific translation (medical, legal)

11 Glossary

Machine Translation (MT)
Automatic conversion of text from one language to another

RBMT
Rule-Based Machine Translation using hand-crafted linguistic rules

SMT
Statistical Machine Translation learning from parallel corpora

NMT
Neural Machine Translation using deep learning

Seq2Seq
Sequence-to-Sequence model with encoder-decoder architecture

Encoder
Component that reads and encodes source sentence

Decoder
Component that generates target sentence

Context Vector
Fixed-length representation of input sequence

Attention
Mechanism allowing decoder to focus on relevant input parts

Transformer
Architecture using only attention, no recurrence

Self-Attention
Attention where sequence attends to itself

Multi-Head Attention
Multiple parallel attention operations

BLEU Score
Automatic metric measuring translation quality

N-gram Precision
Fraction of n-grams matching reference

Brevity Penalty
BLEU component penalizing short translations

Hybrid MT
Systems combining multiple translation approaches

Parallel Corpus
Collection of texts with human translations

Alignment
Correspondence between source and target words

12 Learning Checklist

- ☐ Can explain the differences between RBMT, SMT, and NMT
- ☐ Understand why translation is difficult (ambiguity, structure, idioms, context)
- ☐ Can describe Seq2Seq architecture and its bottleneck problem
- ☐ Understand how attention solves the bottleneck problem
- ☐ Know why Transformers are better than RNN-based models
- ☐ Can explain BLEU score calculation and its limitations
- ☐ Understand the role of hybrid systems in specialized domains