NEURAL MACHINE TRANSLATION

Understanding How Modern MT Systems Work

Thomas Moerman

2025-11-07

COURSE OVERVIEW & DATES

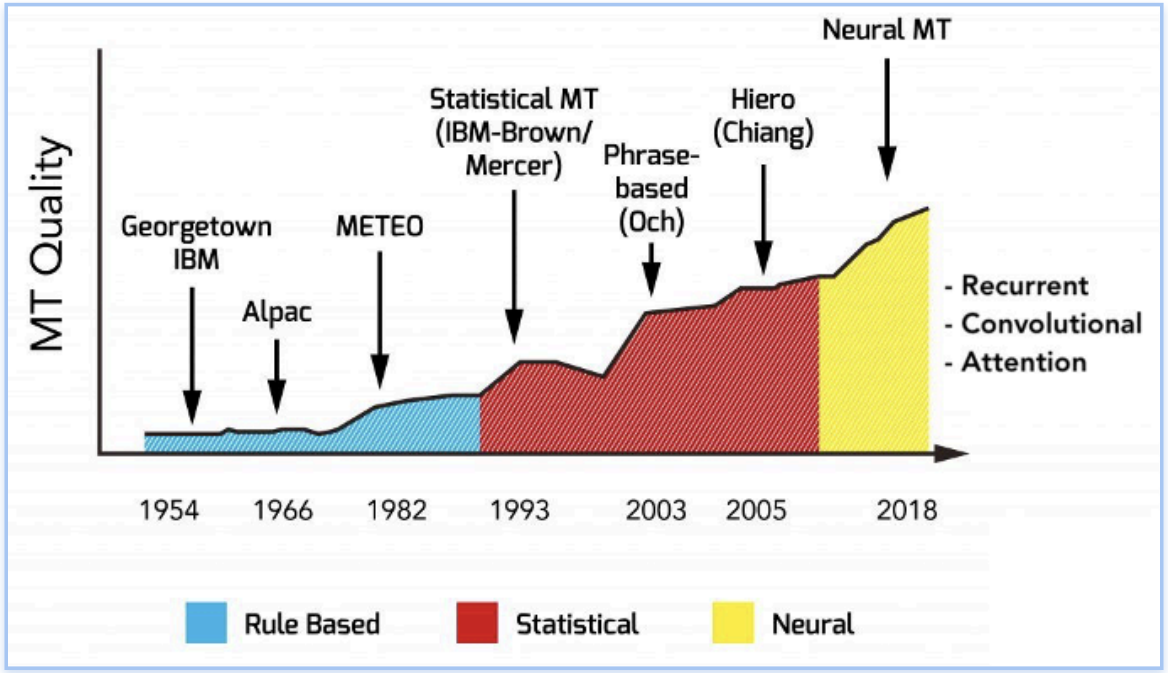
Date	Topic	Lecturer
17/10	Introduction, History, Challenges	Joke Daems
25/10	MT Architectures: RBMT & SMT	Lieve Macken
31/10	MT Evaluation & User Reception	Lieve Macken
07/11	MT Architectures: NMT	Thomas Moerman
14/11	LLMs & MT Evaluation	Thomas Moerman
21/11	Post-editing	Joke Daems
28/11	*No class / Buffer slot*	
05/12	MTPE in Modern Translation Tools	Joke Daems
08/12	Company Visit: Yamagata Europe	
12/12	LLM Applications (*guest lecture: CrossLang*)	

PART 1
INTRODUCTION

WHAT IS NEURAL MACHINE TRANSLATION?

Evolution of Machine Translation:

- 📚 **Rule-based MT** (1950s-1990s): Hand-written grammar rules
- 📊 **Statistical MT** (1990s-2010s): Learn patterns from data
- 🧠 **Neural MT** (2014-present): Deep learning from examples



KEY DIFFERENCE

Traditional MT: Needs explicit rules written by linguists

Neural MT: Learns patterns automatically from thousands or even millions of translation examples

💡 **Like learning a language through immersion vs. studying grammar books**

WHY UNDERSTANDING NMT MATTERS

- 🔍 Better evaluation of MT output quality
- 🎯 Understanding when to trust (or not trust) translations
- 🛠️ Making informed decisions about MT deployment
- 💬 Speaking the same language as developers

OVERVIEW OF TODAY

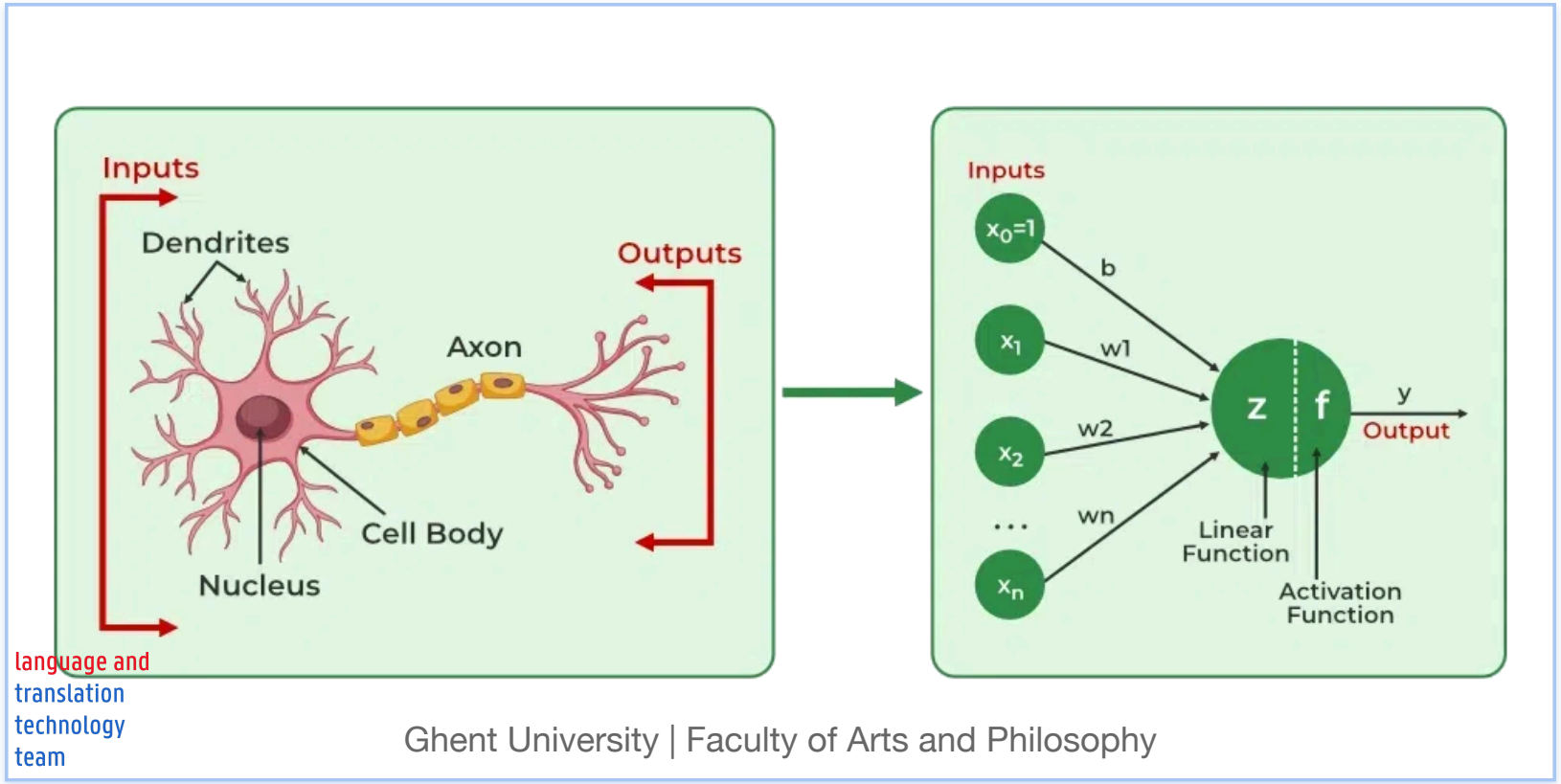
1. Introduction
2. Neural Networks Crash Course
3. From MLPs to Neural Language Models
4. The Transformer and Machine Translation
5. Training and Evaluation of NMT
6. Practical Application (+assignment)

PART 2
NEURAL NETWORKS CRASH COURSE

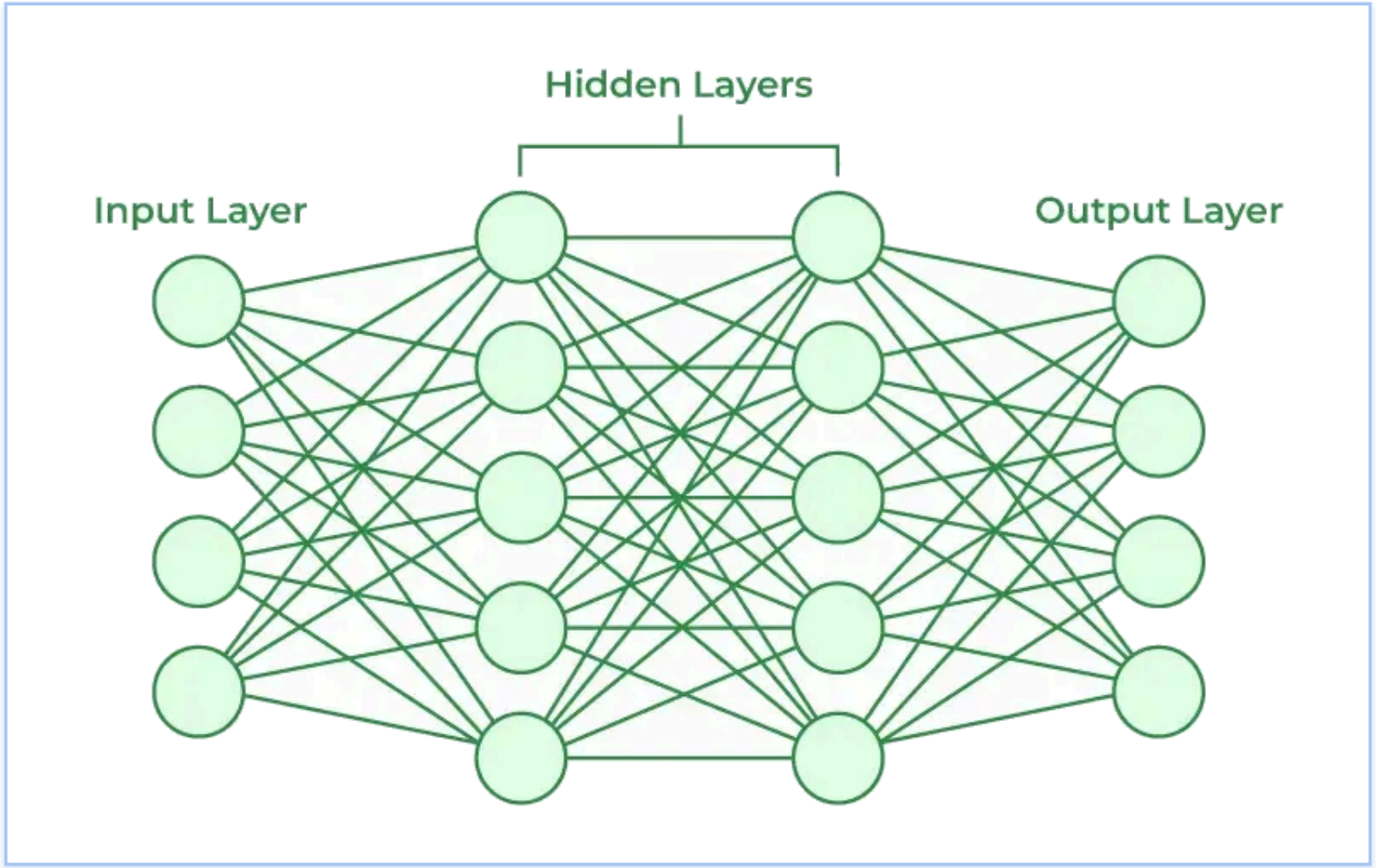
WHAT IS A NEURAL NETWORK?

Simply put: an (artificial) NN is a function that takes:

- a set of inputs
- does some computations on them
- and produces a set of outputs



WHAT IS A NEURAL NETWORK?



Neural network with multiple layers

Source: GeeksforGeeks - Neural Networks: A Beginner's Guide

WHAT IS A NEURAL NETWORK?

Classic example: Neural Networks by 3Blue1Brown

WHAT IS A NEURAL NETWORK?

For example:

Application	Input	Output (Prediction)
Online Advertising	User behavior, demographics	Ad click probability
Image Recognition	Image pixels	Object labels, categories
Spam Detection	Email content, metadata	Spam or not spam
and so on…		

AND OF COURSE: MACHINE TRANSLATION!

Input (Source Language)	Output (Target Language)
“The cat sat on the mat”	“Le chat est assis sur le tapis”

Timeline:

- The paradigm after SMT (Statistical Machine Translation)
- Happened around 2015, with neural systems winning in all language pairs in WMT 2016

AND OF COURSE: MACHINE TRANSLATION!

Similarities to SMT:

- Both are computational approaches to automatic translation
- Both are data-driven methods requiring large parallel corpora
- Both successfully deployed in real-world translation systems

Key Differences from SMT:

- **SMT**: Translates in smaller, independent phrases using statistical probability models and extensive feature engineering
- **NMT**: Translates entire sentences end-to-end using a single deep learning model
- **NMT advantage**: Learns context and long-range dependencies
- **Result**: NMT produces more fluid, contextually aware, human-like output vs. robotic/fragmented SMT

BUILDING BLOCKS: CREATING VALUES

Core idea of NLP is that we can perform operations on words

Let's say we have the following words:

```
[king, queen, man]
```

What we would like to do is something like this:

```
king * queen = royalty
royalty + man = prince
```

BUILDING BLOCKS: CREATING VALUES

What if we assign a value to each word?

```
king = 4.0
queen = 5.0
man = 2.0
```

Now we can perform operations on them:

```
king * queen = royalty # Output: 20.0
royalty + man = prince # Output: 22.0
```

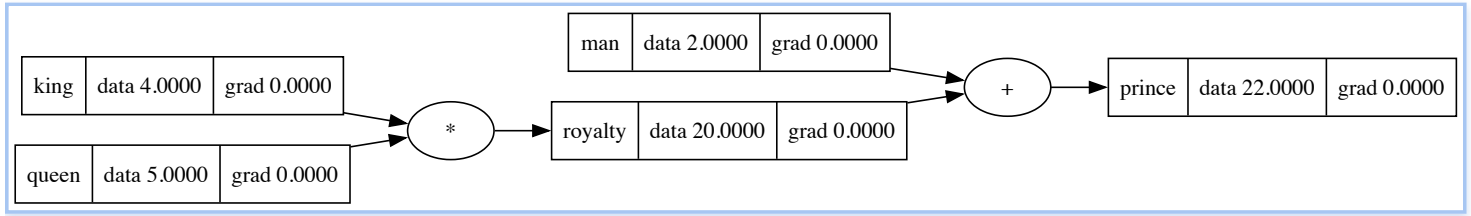
Now we have:

```
king = 4.0
queen = 5.0
man = 2.0
royalty = 20.0
prince = 22.0
```

A NETWORK OF OPERATIONS?

We can represent this as a network of operations:

```
# Visualize the computation graph
draw_dot(prince)
```



LET'S BUILD A SIMPLE NN FROM SCRATCH!

- see notebook

SUMMARY AND WHAT YOU NEED TO REMEMBER

- Foundations of an Artificial Neural Network
  - Operations (Addition, Multiplication, Derivatives/Gradients, Activation Functions)
  - Components (Neurons, Layers, MLP)
  - Forward and Backward Pass
  - Loss

PART 3
FROM MLPS TO NEURAL LANGUAGE MODELS

SO FAR WE HAVE SEEN

✅ Basic neural networks (MLPs):

- Operations (multiply, add, activation functions)
- Forward pass: input → computation → output
- Backward pass: adjust weights to reduce error
- Works great for: fixed-size inputs → fixed-size outputs

❌ **But this is NOT enough for language!**

Example: “The cat sat on the ___”

- Input is a **sequence** of words (variable length!)
- Need to predict **next word** based on context
- Order matters: “cat sat on mat” ≠ “mat on sat cat”

WHAT DO WE NEED FOR A LANGUAGE MODEL?

1. A Task: Next Word Prediction

Given: “The cat sat on the”

Predict: “mat” (or “floor”, “sofa”, etc.)

💡 This is how language models learn language patterns!

2. A way to represent words

- Can’t just use arbitrary numbers (4.0, 5.0, 2.0)
- Need **word embeddings**: vectors that capture meaning
- Similar words → similar vectors
- Example: vector(“king”) ≈ vector(“queen”) and vector(“king”) - vector(“man”) + vector(“woman”) ≈ vector(“queen”)

3. A way to process sequences

- Need to handle **variable-length** input
- Need to remember **context** from earlier words
- Need to understand **order** matters

WHAT DOES THIS LOOK LIKE IN PRACTICE?

- see notebook

BREAK (15 MINUTES)

PART 4
THE TRANSFORMER AND MACHINE TRANSLATION

TRANSFORMER FOR MACHINE TRANSLATION

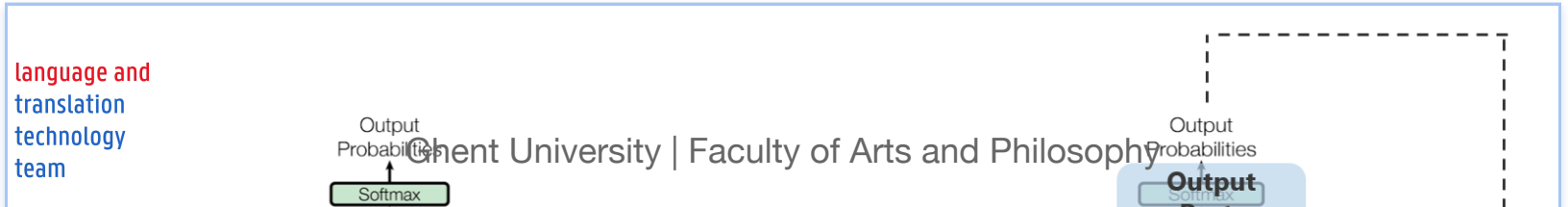
The architecture has two main parts:

Encoder (left side):

- Processes the **source sentence** (e.g., English)
- Each word attends to all other source words
- Builds rich contextual representations
- Output: contextualized source representations

Decoder (right side):

- Generates the **target sentence** (e.g., French)
- Attends to previously generated words (self-attention)
- Attends to source sentence (cross-attention)
- Generates one word at a time



HOW TRANSLATION WORKS: STEP BY STEP

Input sentence (source):

```
<start_input> The cat sat on the mat <end_input>
```

Step 1: Encoding (process entire source at once)

```
Source tokens: [<start_input>, The, cat, sat, on, the, mat, <end_input>]
                        ↓        ↓    ↓    ↓   ↓   ↓    ↓        ↓
Encoder states:      [h0,      h1,  h2,  h3, h4, h5,  h6,      h7]
                              ← contextual representations
```

Each hidden state contains information about the word + its context

HOW TRANSLATION WORKS: STEP BY STEP

Step 2: Decoding (generate one word at a time)

```
Input to decoder                Output from decoder
————————————————————————    ——————————————————
<start_output>                → "Le"
<start_output> Le             → "chat"
<start_output> Le chat        → "est"
<start_output> Le chat est    → "assis"
<start_output> Le ... assis   → "sur"
<start_output> Le ... sur     → "le"
<start_output> Le ... le      → "tapis"
<start_output> Le ... tapis   → <end_output>
```

Stop when `<end_output>` token is generated!

Final output (target):

```
<start_output> Le chat est assis sur le tapis <end_output>
```

(We remove `<start_output>` and `<end_output>` tokens for the user)

THE POWER OF ATTENTION IN TRANSLATION

Example: Translating “The brown fox” → “El zorro marrón”

When generating **“zorro”**:

Source Word	Attention Weight
The	0.05 (low)
brown	0.10 (medium)
fox	0.85 (high)

💡 The model learns **which source words** are relevant for each target word!

WHY TRANSFORMERS REVOLUTIONIZED MT

1. Parallelization

- RNN: processes words one-by-one (slow)
- Transformer: processes all words simultaneously (fast!)
- Training time: days → hours

2. Long-Range Dependencies

- Can connect words far apart in sentence
- Example: “The keys that were on the table **were** lost”
- Subject-verb agreement over long distance

3. Better Context

- Self-attention lets every word see every other word
- Captures nuanced meanings based on full context
- “bank” = financial institution vs. river bank

THE COMPLETE TRANSLATION PIPELINE

Input: “The cat sat on the mat”

1. Tokenization + Add Special Tokens

```
"The cat sat on the mat"
    ↓
["<s>", "_The", "_cat", "_sat", "_on", "_the", "_mat", "</s>"]
```

Note: `_` represents a space (word boundary), `<s>` = start, `</s>` = end

2. Subword Splitting (BPE/SentencePiece)

```
["<s>", "_The", "_cat", "_sat", "_on", "_the", "_mat", "</s>"]
    ↓
["<s>", "_The", "_cat", "_sat", "_on", "_the", "_mat", "</s>"]
                        (common words stay whole)

Rare word example: "demystifying" → ["_de", "myst", "ify", "ing"]
```

THE COMPLETE TRANSLATION PIPELINE

3. Convert to IDs (vocabulary lookup)

```
["<s>", "_The", "_cat", "_sat", ...]
    ↓
[0, 145, 892, 1203, 67, 145, 2341, 1]
```

4. Encoding

```
[0, 145, 892, 1203, 67, 145, 2341, 1]
    ↓ (Encoder: self-attention)
[h0, h1, h2, h3, h4, h5, h6, h7]  ← contextual representations
```

5. Decoding (auto-regressive)

```
<s> → "_Le" → "_chat" → "_est" → "_assis" → "_sur" → "_le" → "_tapis" → </s>
  ↑ Cross-attention to encoder outputs at each step
```

THE COMPLETE TRANSLATION PIPELINE

6. Convert IDs back to tokens

```
[2, 234, 567, 891, 432, 789, 123, 456, 3]
    ↓
["<s>", "_Le", "_chat", "_est", "_assis", "_sur", "_le", "_tapis", "</s>"]
```

7. Remove special tokens + Detokenization

```
["<s>", "_Le", "_chat", "_est", "_assis", "_sur", "_le", "_tapis", "</s>"]
    ↓ (remove <s> and </s>)
["_Le", "_chat", "_est", "_assis", "_sur", "_le", "_tapis"]
    ↓ (replace _ with spaces)
"Le chat est assis sur le tapis"  ← Final output!
```

DECODING STRATEGIES

What is decoding?

The process of generating the target translation from the model's predictions

The challenge: At each step, the model outputs probabilities for 30,000+ possible next words!

How do we choose?

GREEDY DECODING: THE SIMPLE APPROACH

Strategy: Always pick the word with the highest probability

```
Step 1: <s> → probabilities → pick "_Le" (0.87)
Step 2: <s> _Le → probabilities → pick "_chat" (0.76)
Step 3: <s> _Le _chat → probabilities → pick "_est" (0.82)
...
```

Advantages:

- ✅ Fast (one choice per step)
- ✅ Simple to implement

Problem:

- ❌ Locally optimal ≠ globally optimal!
- ❌ May miss better translations by committing too early

THE PROBLEM WITH GREEDY DECODING

Example: French determiners

Source: “I see a cat”

```
Greedy choice at step 1:
  "Je" (0.6) ✓ chosen
  "J'" (0.35) × ignored

But what comes next?
  "Je vois" → needs word boundary → awkward
  "J'ai" → no word boundary needed → more natural!
```

💡 **Insight:** Sometimes a lower-probability choice at one step leads to a better overall translation!

BEAM SEARCH: EXPLORING MULTIPLE PATHS

Strategy: Keep top-K candidates (beam size = K) at each step

Instead of one hypothesis, track multiple possibilities

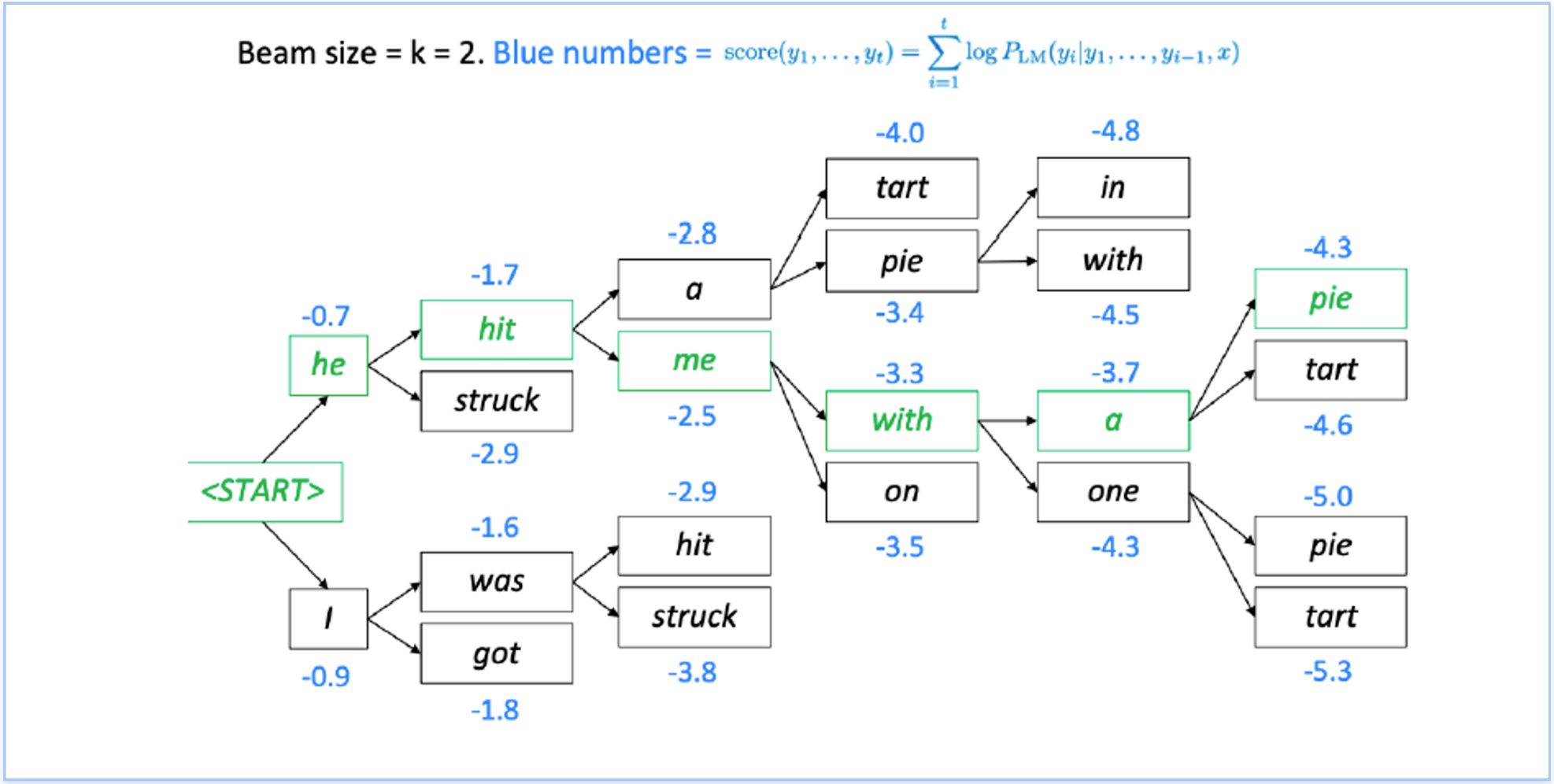
Beam size = 3 example:

```
Step 1: <s> → keep top 3:
  1. "_Le"  (score: -0.14)
  2. "_La"  (score: -0.40)
  3. "_Un"  (score: -0.51)

Step 2: Expand each → keep top 3 overall:
  1. "_Le" + "_chat"    (score: -0.41)
  2. "_Le" + "_matou"   (score: -0.58)
  3. "_La" + "_chatte"  (score: -0.65)

Continue until all reach </s>...
```

BEAM SEARCH: HOW IT WORKS



Beam search explores multiple translation paths simultaneously

BEAM SEARCH: PARAMETERS

Beam size (K):

- Small (K=1): Greedy decoding (fast, risky)
- Medium (K=4-5): Good balance (typical for NMT)
- Large (K=10+): More exploration (slow, diminishing returns)

Length penalty:

- Problem: Shorter sentences often have higher probabilities
- Solution: Normalize scores by length
- Prevents bias toward short translations

Typical settings for NMT:

- Beam size: 4 or 5
- Length penalty: 0.6 - 1.0
- Max length: 1.5-2x source length

N-BEST LISTS: MULTIPLE HYPOTHESES

Result of beam search: Multiple complete translations ranked by score

```
Source: "The cat sat on the mat"

1. "Le chat est assis sur le tapis"        (score: -2.34) ← best
2. "Le chat était assis sur le tapis"      (score: -2.67)
3. "Le chat s'est assis sur le tapis"      (score: -2.89)
4. "Un chat est assis sur le tapis"        (score: -3.12)
5. "Le chat assis sur le tapis"            (score: -3.45)
```

Why n-best lists matter: - See alternative translations - Understand model's uncertainty - Post-processing and reranking - Human translators can choose best option

DECODING IN PRACTICE

Trade-offs:

Aspect	Greedy	Beam (K=5)
Speed	Very fast	~5x slower
Quality	Good	Better
Diversity	None	Multiple options
Memory	Minimal	K× more

When to use what?

- **Greedy:** Real-time applications, speed critical
- **Beam search:** Production MT systems, quality critical
- **Large beams:** Research, reranking experiments

For your assignment: You'll see n-best lists and analyze different hypotheses!

PART 5
TRAINING AND EVALUATION OF NMT

TRAINING

- see notebook

TRAINING SUMMARY

What is training?

- Given a dataset of input-output (source-target) examples, the goal is to find weights that make the model's outputs as close as possible to the targets.
- A **loss function** measures how far the predictions are from the correct answers.
- A **training algorithm** (like gradient descent) makes small adjustments to the weights based on the loss, using a batch of examples.
- An **epoch** = one full pass over the training data.
- The **learning rate** controls how big each update is.

When does training stop?

- We want the model to generalize to new, unseen examples (not just memorize training data).
- If we train too long, the model may **overfit**—memorizing and failing to translate new sentences well.
- **Solution:** Monitor performance on a separate **validation set** and stop training when it stops improving (*early stopping*).

EVALUATION: HOW DO WE KNOW IF TRANSLATION IS GOOD?

The fundamental question: Is the translation correct?

The problem:

- Translation is subjective—many correct translations exist
- Manual evaluation is slow and expensive
- Need quick feedback during development

Solutions:

- **Human evaluation:** Gold standard but expensive
- **Automatic metrics:** Fast approximations for development

HUMAN EVALUATION: THE GOLD STANDARD

Direct Assessment (current best practice):

- Evaluators rate translations on continuous scale (0-100)
- Can be **monolingual** (with reference) or **bilingual** (with source)
- Scores normalized per evaluator to handle inconsistencies

Human Translation Edit Rate (HTER):

- Measure words changed during postediting
- Reflects actual correction effort
- More objective than subjective ratings

The challenge: Human evaluation requires many evaluators, costs money, takes time

Solution for development: Automatic metrics!

AUTOMATIC METRICS: FAST APPROXIMATIONS

Key idea: Compare MT output to reference translation(s)

- More similar to reference → higher score
- Enables rapid testing during development

Traditional metrics (string-based):

- BLEU, METEOR, TER, chrF
- Count matching words/characters

Neural metrics (embedding-based):

- COMET, BERTScore
- Understand meaning, not just strings

BLEU: THE DOMINANT METRIC

BLEU = Bilingual Evaluation Understudy

How it works:

1. Count matching n-grams (1-4 words) between MT and reference
2. Compute precision for each n-gram size
3. Combine with geometric mean
4. Apply brevity penalty for short outputs

Score range: 0-100 (higher is better)

- 0 = no overlap with reference
- 100 = perfect match with reference

BLEU: EXAMPLE CALCULATION

Reference: “The cat is sitting on the mat”

MT output: “The cat sits on the mat”

1-gram matches: The, cat, on, the, mat (5/6 = 83.3%)

2-gram matches: “The cat”, “on the”, “the mat” (3/5 = 60.0%)

3-gram matches: “on the mat” (1/4 = 25.0%)

4-gram matches: none (0/3 = 0%)

Geometric mean: $\sqrt[4]{(0.833 \times 0.60 \times 0.25 \times 0.01)} \approx 0.13$

Brevity penalty (length okay) = 1.0

Final BLEU: ~13

INTERPRETING BLEU SCORES

Important: Ranges vary by language pair and domain!

- Easy pairs (EN↔FR): higher scores
- Difficult pairs (EN↔Chinese): lower scores

BLEU: STRENGTHS AND WEAKNESSES

Strengths:

- ✅ Simple and fast to compute
- ✅ Correlates reasonably with human judgment (at corpus level)
- ✅ Multiple references help account for variation
- ✅ Widely used → easy to compare systems
- ✅ Language-independent (no linguistic tools needed)

BLEU: STRENGTHS AND WEAKNESSES

Weaknesses:

- ❌ Ignores meaning and semantics
- ❌ Treats all words equally (“not” = “the”)
- ❌ Poor at sentence-level evaluation
- ❌ Rewards n-gram matches, not translation quality
- ❌ Absolute scores are meaningless (only comparisons matter)
- ❌ Requires reference translations

OTHER TRADITIONAL METRICS

TER (Translation Edit Rate):

- Counts edits needed to fix translation (insertions, deletions, substitutions, shifts)
- Score = percentage of words needing change
- **Interpretation:** TER = 30% means 30% of words need editing
- ✅ More intuitive than BLEU
- ❌ Edit distance doesn't capture semantics

OTHER TRADITIONAL METRICS

chrF (Character n-gram F-score):

- Character-level instead of word-level
- ✅ Better for morphologically rich languages
- ✅ Partial credit for similar words (“running” vs “runs”)
- ❌ More sensitive to minor spelling differences
- ❌ Can overestimate similarity for sentences that only share roots or affixes

METEOR:

- Uses stemming and synonyms (via WordNet)
- ✅ More sophisticated than BLEU
- ❌ Requires language-specific resources
- ❌ Never widely adopted

THE RISE OF NEURAL METRICS

Problem with traditional metrics:

- Only reward exact string matches
- Miss semantically equivalent translations

Example where traditional metrics fail:

Reference: “The car is red”

MT1: “The automobile is red” ← semantically correct

MT2: “The red is car” ← grammatically wrong

BLEU treats both similarly (50% word overlap) but humans strongly prefer MT1!

COMET: NEURAL METRIC

COMET = Crosslingual Optimized Metric for Evaluation of Translation

Key innovation: Uses neural networks to estimate translation quality

How it works (using principles we've learned!):

1. **Encoder:** Pre-trained multilingual transformer (e.g., XLM-R)
   - Encodes source, translation, and reference into vector representations
   - Uses same attention mechanisms we saw in Part 3!
2. **Regression head:** Neural network predicts quality score
   - Trained on human judgments (Direct Assessment scores)
   - Learns what makes translations good/bad

COMET: TRAINING AND ARCHITECTURE

Training data:

- Thousands of (source, MT, reference, human_score) examples
- From WMT evaluation campaigns (human judgments)

What it learns:

- Semantic similarity (not just string matching)
- Fluency and grammaticality
- Adequacy (meaning preservation)
- What human evaluators care about

Score range: Typically -1 to +1 (higher is better)

- Positive scores: good translations
- Negative scores: poor translations
- Unlike BLEU, scores are learned from human behavior!

COMET EXAMPLE

Reference: “The car is red”

MT1: “The automobile is red” ← synonym

MT2: “The red is car” ← wrong word order

Traditional metric (BLEU):

- Both get similar scores (~50% overlap)

COMET:

- MT1: +0.85 (high score) ← understands “car” ≈ “automobile”
- MT2: -0.42 (low score) ← understands grammar is wrong

💡 COMET captures meaning, not just word overlap!

NEURAL METRICS: STRENGTHS AND WEAKNESSES

Strengths:

- ✅ Capture semantic similarity (synonyms, paraphrases)
- ✅ Better correlation with human judgment
- ✅ Understand context and meaning
- ✅ Can work without references (quality estimation mode)
- ✅ Trained on actual human preferences

NEURAL METRICS: STRENGTHS AND WEAKNESSES

Weaknesses:

- ❌ Computationally expensive (requires GPU)
- ❌ Less interpretable (“black box”)
- ❌ Require large amounts of training data
- ❌ May inherit biases from training data
- ❌ Newer → less established trust in research community

METRIC COMPARISON: CORRELATION WITH HUMANS

How metrics are evaluated: Correlation with human judgments

Higher correlation = better metric

Typical correlations (segment-level):

Metric	Correlation with Humans
BLEU	~0.30 (weak)
TER	~0.40 (moderate)
chrF	~0.50 (moderate)
METEOR	~0.45 (moderate)
COMET	**~0.75 (strong)**
BERTScore	~0.65 (strong)

Takeaway: Neural metrics significantly outperform traditional metrics, especially at sentence level!

WHICH METRIC SHOULD YOU USE?

BLEU remains dominant because:

- Historical standard (everyone uses it)
- Fast and simple
- Doesn't require training data or GPU
- Interpretability concerns with neural metrics
- Research inertia

Best practice (2024):

- **BLEU:** For quick development feedback and comparing to published results
- **COMET:** For more accurate quality assessment and final evaluation
- **Human evaluation:** For definitive quality claims and publication

For your assignment:

- You'll work with BLEU (standard in NMT research)
- Focus on understanding trends and comparisons
- Remember: absolute scores don't matter, only comparisons!

EVALUATION: KEY TAKEAWAYS

Important principles:

1. **No perfect metric exists** – translation quality is subjective
2. **Use metrics for comparison**, not absolute assessment
3. **Automatic metrics ≠ human judgment** – they're approximations
4. **Different metrics measure different things** (fluency vs adequacy)
5. **Neural metrics are better but less interpretable**
6. **Human evaluation is the gold standard**

EVALUATION: KEY TAKEAWAYS

For your work:

- Understand what BLEU measures (n-gram overlap)
- Know its limitations (ignores meaning)
- Use it to track improvement, not prove “human parity”
- Always interpret scores in context (language pair, domain)

CHALLENGES IN NMT

Common challenges:

- **Out-of-domain data:** NMT often performs worse than SMT when translating text that differs from its training data.
- **Low-resource settings:** NMT struggles with languages or domains where only small amounts of parallel data are available.
- **Rare words:** NMT has difficulty translating very infrequent or unseen words.
- **Interpretability:** It is harder to understand or explain NMT decisions compared to SMT.
- **Efficiency:** NMT models are typically less efficient—requiring more computation and resources than SMT systems.

PART 6
PRACTICAL APPLICATION

ASSIGNMENT EXPECTATIONS

What to do:

1. ✅ Analyze training logs (find plateau)
2. ✅ Test difficult sentences
3. ✅ Examine n-best lists
4. ✅ Explain observations with proper terminology

What NOT to do:

- ❌ Just describe without explaining WHY
- ❌ Ignore the concepts from class

COMMON PITFALLS

❌ **“The BLEU score goes up then down”**

✅ “BLEU increases until update 6000, then plateaus, indicating the model has converged and further training risks overfitting”

❌ **“The translation is bad”**

✅ “The system struggles with ‘ethylene copolymer’ because this technical term is rare in the TED talk training data, causing poor subword segmentation”

USING PROPER TERMINOLOGY

Terms that can be used:

- Epoch, batch, step/update
- Loss, BLEU, validation
- Encoder, decoder, attention
- Overfitting, plateau, convergence
- Beam search, n-best list
- Domain mismatch, vocabulary coverage

Speak the language! 🗣️

PART 7
WRAP-UP

KEY TAKEAWAYS

Neural networks:

- Learn patterns from examples
- Adjust weights through training
- Multiple layers refine information

NMT architecture:

- Encoder “understands” source
- Decoder generates translation
- Attention focuses on relevant words

QUESTIONS?

Topics we covered:

- Neural networks basics
- Encoder-decoder architecture
- Training process and metrics
- Reading logs and analyzing output

Need clarification on anything?

NEXT WEEK: LLMS IN TRANSLATION