Lecture 20: Applications of Encoder Models

Week 6, Lecture 3 - From Theory to Practice

PSYC 51.07: Models of Language and Communication

Winter 2026

Today's Agenda 📋

1. 🚀 **Real-World Applications**: Where BERT shines
2. 🧠 **Cognitive Neuroscience**: Brain-model parallels
3. 🤔 **Understanding vs. Pattern Matching**: The big debate
4. ⚠️ **Limitations**: What BERT can't do
5. 💡 **Practical Tips**: Deployment and optimization
6. 🔮 **Future Directions**: Where are we heading?

Goal: Connect BERT to real applications and understand broader implications

BERT Applications 🚀

BERT excels at understanding tasks:

Classification Tasks:

- Sentiment Analysis
- Topic Classification
- Spam Detection
- Intent Recognition

Token-Level Tasks:

- Named Entity Recognition (NER)
- Part-of-Speech Tagging
- Word Sense Disambiguation

Case Study: Google Search 🔍

BERT revolutionized search in 2019

```
# Why word order matters: BERT understands prepositions!
query = "2019 brazil traveler to usa need a visa"

# Before BERT (bag-of-words matching):
keywords = ["brazil", "traveler", "usa", "visa"]
# Matches both: "US traveler to Brazil" AND "Brazil traveler to US"

# With BERT (contextual understanding):
bert_understanding = {
    "subject": "brazil traveler",      # WHO is traveling
    "destination": "usa",               # WHERE they're going
    "direction": "brazil → usa",        # The preposition "to" is key!
    "intent": "visa requirements"
}
# BERT correctly ranks: "Brazil citizen visa requirements for USA"
```

Question Answering with BERT 💬

Extractive QA: Find answer span in passage

Example

Context: "The Normans (Norman: Nourmands; French: Normands; Latin: Normanni) were the people who in the 10th and 11th centuries gave their name to Normandy, a region in France."

Question: "In what country is Normandy located?"

Answer: France

```
from transformers import pipeline

# Load QA pipeline with BERT
qa_pipeline = pipeline("question-answering", model="bert-large-uncased-whole-wo

```

Named Entity Recognition 🏷️

Token-level classification task

Example

Input: "Apple Inc. is headquartered in Cupertino, California."

Output:

- Apple Inc. → {ORGANIZATION}
- Cupertino → {LOCATION}
- California → {LOCATION}

```
from transformers import pipeline

# Load NER pipeline
ner_pipeline = pipeline("ner", model="dslim/bert-base-NER")
```

Sentiment Analysis 😊😐😢

Sequence classification task

Examples

- "This movie was absolutely amazing!" → {POSITIVE}
- "The product broke after one week." → {NEGATIVE}
- "The weather is cloudy today." → {NEUTRAL}

```
from transformers import pipeline

# Load sentiment analysis pipeline
sentiment_pipeline = pipeline("sentiment-analysis", model="distilbert-base-unca

# Analyze sentiments
texts = [
    "This movie was absolutely amazing!",
    "The product broke after one week."
```

Semantic Similarity 🔗

Measuring sentence similarity with BERT embeddings

```
from transformers import BertTokenizer, BertModel
import torch
import torch.nn.functional as F

tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
model = BertModel.from_pretrained('bert-base-uncased')

def get_sentence_embedding(sentence):
    inputs = tokenizer(sentence, return_tensors='pt', padding=True, truncation=
    outputs = model(**inputs)
    # Use [CLS] token embedding as sentence representation
    return outputs.last_hidden_state[:, 0, :]

# Compare sentences
sent1 = "The cat is sleeping on the couch"
sent2 = "A feline is resting on the sofa"
```

Semantic Similarity 🔗

```
emb3 = get_sentence_embedding(sent3)

# Compute cosine similarities
sim_12 = F.cosine_similarity(emb1, emb2).item()
sim_13 = F.cosine_similarity(emb1, emb3).item()

print(f"Similarity (1-2): {sim_12:.3f}")  # High (paraphrases)
print(f"Similarity (1-3): {sim_13:.3f}")  # Low (different topics)
```

...continued

Cognitive Neuroscience Perspective 🧠

How do brains and models process language?

Predictive Processing in the Brain:

- Brain constantly predicts upcoming input
- N400: Neural response to unexpected words
- P600: Syntactic anomaly detection
- Context shapes predictions
- Prediction errors drive learning

Key Brain Regions:

- **Left IFG**: Syntax processing
- **Left STG/MTG**: Semantic processing

Prediction in Brains vs. Language Models 🧠🤖

Parallels between neural and artificial systems

Phenomenon	Human Brain	Transformer Models
Surprise	N400 amplitude (EEG)	Cross-entropy loss
Hierarchy	sounds → words → sentences	tokens → phrases → meaning
Context	Prior discourse, world knowledge	Self-attention over sequence
Representation	Population coding (neurons)	Distributed embeddings (vectors)

```
# Concrete example: Surprise/N400 parallel
sentence_a = "I take my coffee with cream and sugar"  # Expected
sentence_b = "I take my coffee with cream and socks"  # Surprising

# Brain: N400 amplitude higher for "socks"
```

Neural Encoding with Language Models 🔬

Can we predict brain activity from language models?

```
# Neural encoding experiment workflow
import numpy as np
from transformers import BertModel

# 1. Participant reads sentences while in fMRI scanner
sentences = ["The dog chased the cat", "She opened the door", ...]
brain_activity = fmri_scanner.record(sentences)  # (n_sentences, n_voxels)

# 2. Extract BERT representations for same sentences
bert = BertModel.from_pretrained("bert-base-uncased")
bert_embeddings = []
for sent in sentences:
    outputs = bert(tokenizer(sent, return_tensors="pt"))
    # Use layer 8 (found to correlate best with semantic areas)
    bert_embeddings.append(outputs.hidden_states[8].mean(dim=1))

```

Neural Encoding with Language Models 🔬

```
# 4. Predict brain activity for new sentences
predictions = encoder.predict(bert_embeddings[80:])
correlation = np.corrcoef(predictions.flat, brain_activity[80:].flat)[0,1]
# Correlation ~ 0.3-0.5 in language areas (significant!)
```

...continued

Key finding: BERT layer 8 best predicts semantic areas; layers 2-4 predict phonological areas

Reference: Caucheteux & King (2022) - "Brains and algorithms partially converge"

Discussion: What Does the Model "Understand"? 🤔

Does BERT understand language?

Evidence FOR understanding:

- Captures syntax and semantics
- Resolves ambiguity
- Handles long-range dependencies
- Generalizes to new examples
- Predicts brain activity
- Solves complex tasks

"If it acts like it understands, maybe it does?"

Evidence AGAINST understanding:

Adversarial Examples and Brittleness ⚠️

BERT can be fooled easily

```
from transformers import pipeline
classifier = pipeline("sentiment-analysis")

# Works correctly
classifier("This movie was absolutely wonderful!")
# → [{'label': 'POSITIVE', 'score': 0.9998}]

# Adding irrelevant negative words flips prediction!
classifier("This movie was absolutely wonderful! [SEP] bad bad bad bad")
# → [{'label': 'NEGATIVE', 'score': 0.9234}]  # WRONG!

# Synonym substitution can break it
classifier("The food was good")   # → POSITIVE (0.99)
classifier("The food was fine")   # → POSITIVE (0.72)  # Less confident
classifier("The food was ok")     # → NEGATIVE (0.51)  # WRONG!

```

Limitations of Current Models ⚠️

Despite impressive performance, transformers have limitations:

1. **Quadratic Complexity**
    - Self-attention scales as $O(n^2)$

- Limited context windows (512-4096 tokens)
- Cannot process very long documents efficiently

2. **No True Understanding**
    - Pattern matching vs. comprehension

- Lack of common sense
- No world model

3. **Data Efficiency**
    - Requires massive training data

Bias in Language Models ⚖️

Models reflect and can amplify societal biases

```
from transformers import pipeline
unmasker = pipeline("fill-mask", model="bert-base-uncased")

# Gender bias in occupations
unmasker("The doctor said [MASK] would be late.")
# → [('he', 0.62), ('she', 0.18), ('it', 0.08), ...]

unmasker("The nurse said [MASK] would be late.")
# → [('she', 0.71), ('he', 0.15), ('it', 0.06), ...]

# Racial bias (different sentiment for names)
classifier = pipeline("sentiment-analysis")
classifier("Emily is a brilliant scientist.")  # POSITIVE: 0.98
classifier("Jamal is a brilliant scientist.")  # POSITIVE: 0.94  # Lower!

# Where does bias come from?
```

Practical Tips for Working with Transformers 💡

1. **Start with Pre-trained Models**
    - Don't train from scratch (too expensive!)

- Use HuggingFace Model Hub
- Choose appropriate model size

2. **Fine-tuning Best Practices**
    - Use small learning rate (1e-5 to 5e-5)

- Add warmup steps
- Monitor for overfitting
- Freeze early layers if data is limited

3. **Computational Efficiency**
    - Use mixed precision training (FP16)

Deployment Considerations 🚀

Moving from research to production

```
# Example: Optimizing BERT for production deployment
from transformers import BertModel, BertTokenizer
import torch
import onnxruntime

# Step 1: Load model
model = BertModel.from_pretrained("bert-base-uncased")
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")

# Step 2: Quantize for speed (INT8 instead of FP32)
quantized_model = torch.quantization.quantize_dynamic(
    model, {torch.nn.Linear}, dtype=torch.qint8
)
# Result: 4x smaller, 2x faster on CPU

# Step 3: Export to ONNX for production
```

Deployment Considerations 🚀

```
session = onnxruntime.InferenceSession("bert.onnx")
# 1.5x faster than PyTorch, works on any platform
```

...continued

Optimization	Size	Latency	Quality
Original (FP32)	420MB	50ms	100%
Quantized (INT8)	110MB	25ms	99.5%
ONNX + Quantized	110MB	20ms	99.5%
DistilBERT + ONNX	65MB	12ms	97%

Future Directions 🔮

Where is the field heading?

1. **Longer Context**
    - Efficient attention mechanisms (linear, sparse)

- Models with 100K+ token context
- Better long-document understanding

2. **Multimodal Models**
    - Vision + Language (CLIP, DALL-E)

- Audio + Language (Whisper)
- Grounded understanding

3. **Better Pre-training**
    - More efficient objectives

Encoder vs Decoder Models Revisited 🔄

Different models for different tasks

p{4.5cm}}	Encoder (BERT)	Decoder (GPT)
• Classification		
• NER, QA		
• Similarity	Generation tasks:	
• Text completion		
• Dialogue		
• Creative writing		

Interesting observation:

- Decoder-only models (GPT-3, LLaMA) can also do classification via prompting!

Discussion Questions 💭

1. **Understanding vs. Pattern Matching:**
    - Where do you draw the line?

- Is there a test for "true" understanding?
- Does it matter for applications?

2. **Brain-Model Parallels:**
    - How useful are these comparisons?

- What can neuroscience learn from AI?
- What can AI learn from neuroscience?

3. **Bias and Fairness:**
    - Who is responsible for addressing bias?

- Can we ever have completely unbiased models?

Assignment 4: Context-Aware Models 📝

Hands-on experience with transformers!

Tasks:

1. **Implement Attention Mechanism**
    - Build scaled dot-product attention from scratch

- Visualize attention weights

2. **Fine-tune BERT**
    - Load pre-trained BERT

- Fine-tune on sentiment analysis
- Compare to baseline models

3. **Analyze Contextual Embeddings**

Summary: Weeks 5-6 🎯

What we learned:

1. **Evolution of Context**
    - Seq2Seq → Attention → Transformers

- From bottleneck to full parallelization

2. **Transformer Architecture**
    - Self-attention, multi-head attention

- Positional encoding, layer norm, residuals
- Encoder-only (BERT), Decoder-only (GPT), Both (T5)

3. **BERT & Variants**
    - Masked Language Modeling

Resources & Further Reading 📚

Key Papers:

- Vaswani et al. (2017) - Attention Is All You Need
- Devlin et al. (2019) - BERT: Pre-training of Deep Bidirectional Transformers
- Liu et al. (2019) - RoBERTa
- Sanh et al. (2019) - DistilBERT
- Dao et al. (2022) - FlashAttention

Cognitive Neuroscience:

- Hagoort & Indefrey (2014) - The neurobiology of language beyond single words
- Kuperberg & Jaeger (2016) - What do we mean by prediction in language comprehension?
- Willems et al. (2016) - Prediction during natural language comprehension

Looking Forward in the Course 🔮

Where do we go from here?

Upcoming Topics:

- **Week 7:** Decoder models and text generation (GPT family)
- **Week 8:** Scaling laws and large language models
- **Week 9:** Prompting, in-context learning, and instruction tuning
- **Week 10:** Alignment, RLHF, and ethical considerations

The Journey Continues:

- From understanding (BERT) to generation (GPT)
- From supervised learning to few-shot learning
- From narrow tasks to general-purpose models

Questions? 🙋

Discussion Time

Topics to discuss:

- BERT applications
- Understanding vs. pattern matching
- Cognitive neuroscience connections
- Limitations and future work
- Assignment 4 questions

Thank you! 🙏

See you in Week 7 for GPT and text generation!