Inhaltsverzeichnis

Research Paper Assignment - Week 10

Can Perplexity Detect AI-Generated Text?

Overview

Research Question: “Can perplexity be used as a metric to distinguish AI-generated text from human-written text?”

Goal: Learn what perplexity is, how to compute it, and investigate whether it can serve as an indicator for AI-generated content.

Deliverables: 1. Research paper (3-4 pages PDF) 2. Jupyter notebook with experiments

Due: 1.5 weeks from assignment date

Why this matters: AI-generated text detection is a hot topic in academia, journalism, and content moderation. Understanding perplexity helps you understand how language models work!

Learning Objectives

By completing this assignment, you will:

1. **Understand** what perplexity measures and why it matters
2. **Compute** perplexity using a pre-trained language model
3. **Compare** perplexity distributions of human vs. AI text
4. **Apply statistical tests** (t-test, confidence intervals, effect size)
5. **Evaluate** perplexity as a detection metric (ROC curve, threshold)
6. **Discuss** limitations and real-world applicability

Part 1: Background - What is Perplexity?

Intuition

Perplexity measures how “surprised” a language model is by a text.

- **Low perplexity:** The model expected this text (predictable)
- **High perplexity:** The model did NOT expect this text (surprising)

Formula

```
Perplexity = exp(−1/N × Σ log P(token_i | context))
```

In simple terms: - For each word, the model predicts a probability - If probabilities are high □ low perplexity - If probabilities are low □ high perplexity

The Detection Hypothesis

Hypothesis: AI-generated text has LOWER perplexity than human text.

Why? Because AI generates text that it considers likely. Human text is more varied, creative, and unpredictable.

```
AI Text:    "The weather is nice today." → Model: "Yes, I'd write that!" → Low PPL
Human Text: "The sky wept silver tears." → Model: "Unexpected!"          → High PPL
```

Required Reading

Before starting, skim: - Mitchell et al. (2023): “DetectGPT: Zero-Shot Machine-Generated Text Detection” - Available: https://arxiv.org/abs/2301.11305 - Read: Abstract, Section 1, Section 2.1

Part 2: Experimental Setup

Data Collection

You need TWO sets of texts:

Set A: Human-Written Texts (20-30 samples)

Sources (choose one or mix): - Wikipedia article paragraphs - News articles (BBC, Reuters) - Book excerpts (Project Gutenberg) - Student essays or blog posts - MUST!!!! : Take your own papers from Assignment 3 were you have written the text.

Set B: AI-Generated Texts (20-30 samples)

Generate using: - ChatGPT, Claude, or similar - Use prompts like: “Write a paragraph about [topic]” - Match topics to your human texts for fair comparison!

Requirements: - Each text: 50-150 words - Similar topics across both sets - No cherry-picking (use all generated texts)

Computing Perplexity

Use GPT-2 from Hugging Face (runs locally, free): You can use other models as long as your GPU is able to run it. (7b models would be good, to get better results). If you want to use the Univerity GPU Ressources, you can ask us for access.

```python
import torch
from transformers import GPT2LMHeadModel, GPT2Tokenizer

# Load model
model = GPT2LMHeadModel.from_pretrained('gpt2')
tokenizer = GPT2Tokenizer.from_pretrained('gpt2')
model.eval()

def calculate_perplexity(text):
    """Calculate perplexity of a text using GPT-2."""
    encodings = tokenizer(text, return_tensors='pt')

    with torch.no_grad():
        outputs = model(**encodings, labels=encodings['input_ids'])
        loss = outputs.loss
        perplexity = torch.exp(loss).item()

    return perplexity

# Example usage
text = "The quick brown fox jumps over the lazy dog."
ppl = calculate_perplexity(text)
print(f"Perplexity: {ppl:.2f}")
```

Part 3: Required Experiments

Experiment 1: Perplexity Distribution Comparison with Statistical Testing

Goal: Compare perplexity of human vs. AI texts AND test if the difference is statistically significant

Steps: 1. Calculate perplexity for all human texts 2. Calculate perplexity for all AI texts 3. Compute descriptive statistics (mean, std, min, max) 4. Compute 95% confidence intervals for both means 5. Perform independent samples t-test 6. Calculate effect size (Cohen's d) 7. Create visualization (box plot or histogram)

Table 1: Descriptive Statistics

Source	n	Mean PPL	Std Dev	95% CI	Min	Max
Human	XX	XX.X	XX.X	[XX.X, XX.X]	XX.X	XX.X
AI-Generated	XX	XX.X	XX.X	[XX.X, XX.X]	XX.X	XX.X

Table 2: Statistical Test Results

Test	Statistic	p-value	Interpretation
Independent t-test	t = X.XX	p = X.XXX	Significant? Yes/No
Cohen's d	d = X.XX	-	Effect size: small/medium/large

Interpretation guide for Cohen's d: - |d| < 0.2: Negligible effect - |d| 0.2 - 0.5: Small effect - |d| 0.5 - 0.8: Medium effect - |d| > 0.8: Large effect

Figure 1: Box plot comparing distributions (include means with 95% CI error bars)

Analysis Questions: - Is there a statistically significant difference (p < 0.05)? - How large is the effect (Cohen's d)? - Do the confidence intervals overlap? - How much overlap exists in the distributions?

Code for Statistical Analysis:

```python
import numpy as np
from scipy import stats

# Calculate confidence interval
def confidence_interval(data, confidence=0.95):
    n = len(data)
    mean = np.mean(data)
    se = stats.sem(data)  # Standard error
    h = se * stats.t.ppf((1 + confidence) / 2, n - 1)
    return mean - h, mean + h

# Independent samples t-test
t_stat, p_value = stats.ttest_ind(human_ppls, ai_ppls)

# Cohen's d (effect size)
def cohens_d(group1, group2):
    n1, n2 = len(group1), len(group2)
    var1, var2 = np.var(group1, ddof=1), np.var(group2, ddof=1)
    pooled_std = np.sqrt(((n1-1)*var1 + (n2-1)*var2) / (n1+n2-2))
    return (np.mean(group1) - np.mean(group2)) / pooled_std

effect_size = cohens_d(human_ppls, ai_ppls)

# Results
print(f"Human PPL: {np.mean(human_ppls):.2f} ± {np.std(human_ppls):.2f}")
print(f"AI PPL: {np.mean(ai_ppls):.2f} ± {np.std(ai_ppls):.2f}")
print(f"95% CI Human: {confidence_interval(human_ppls)}")
print(f"95% CI AI: {confidence_interval(ai_ppls)}")
print(f"t-statistic: {t_stat:.3f}")
print(f"p-value: {p_value:.4f}")
print(f"Cohen's d: {effect_size:.3f}")
print(f"Significant difference: {'Yes' if p_value < 0.05 else 'No'}")
```

Experiment 2: Classification Performance

Goal: Evaluate perplexity as a binary classifier

Steps: 1. Combine all texts with labels (0=AI, 1=Human) 2. Use perplexity as the “score” 3. Calculate ROC curve and AUC 4. Find optimal threshold 5. Report accuracy, precision, recall at threshold

```
from sklearn.metrics import roc_curve, auc, accuracy_score

# Prepare data
perplexities = human_ppls + ai_ppls
labels = [1]*len(human_ppls) + [0]*len(ai_ppls)  # 1=human, 0=AI

# ROC curve (higher PPL → more likely human)
fpr, tpr, thresholds = roc_curve(labels, perplexities)
roc_auc = auc(fpr, tpr)

# Find best threshold
optimal_idx = (tpr - fpr).argmax()
optimal_threshold = thresholds[optimal_idx]
```

Table 2: Classification Performance

Metric	Value
ROC-AUC	X.XX
Optimal Threshold	XX.X
Accuracy	XX.X%
Precision (Human)	XX.X%
Recall (Human)	XX.X%

Figure 2: ROC curve with AUC score

Experiment 3: Error Analysis

Goal: Understand when perplexity fails

Steps: 1. Identify misclassified texts (at optimal threshold) 2. Analyze 2-3 false positives (AI classified as human) 3. Analyze 2-3 false negatives (human classified as AI) 4. Discuss patterns

Questions to answer: - What makes some AI text have high perplexity? - What makes some human text have low perplexity? - Are there systematic failure patterns?

Part 4: Paper Structure (3-4 pages)

1. Abstract (100-150 words)

Brief summary: research question, method, key findings, conclusion.

---

2. Introduction (0.5 page)

- Why AI text detection matters (1-2 sentences)
- What is perplexity? (brief explanation)
- Your hypothesis
- Paper overview

---

3. Method (0.75-1 page)

3.1 Data Collection - Human text sources and count - AI text generation method and count - Text length and topic matching

3.2 Perplexity Computation - Model used (GPT-2) - Computation method - Any preprocessing

3.3 Evaluation Approach - Metrics: ROC-AUC, accuracy - Threshold selection method

---

4. Results (1-1.25 pages)

4.1 Perplexity Distributions - Table 1: Descriptive statistics with 95% confidence intervals - Figure 1: Box plot with error bars - Interpretation of distributions

4.2 Statistical Analysis - Table 2: t-test results (t-statistic, p-value) - Cohen's d effect size and interpretation - Are confidence intervals overlapping? - Conclusion: Is the difference statistically significant?

4.3 Classification Performance - Table 3: ROC-AUC, accuracy, precision, recall - Figure 2: ROC curve - Interpretation

4.4 Error Analysis - Example misclassifications - Patterns observed

---

5. Discussion (0.5-0.75 page)

Address: - Does perplexity work as a detector? How well? - Why does it work / not work? - Limitations: - Small dataset - Single model (GPT-2) - Specific text types - Easy to circumvent? - Practical implications

---

6. Conclusion (0.25 page)

- Answer research question
- Main takeaway
- One sentence on future work

---

7. References

Minimum 3-4 references: 1. Mitchell et al. (2023) - DetectGPT 2. GPT-2 paper or Hugging Face documentation 3. One paper on AI text detection or perplexity 4. Any data sources used

---

Part 5: Evaluation Rubric

Total: 100 points

1. Data Collection & Preparation (15 points)

- Human text collection appropriate and diverse (5 pts)
- AI text generation systematic and documented (5 pts)
- Topic matching and text length requirements met (5 pts)

2. Methodology & Perplexity Computation (20 points)

- Perplexity correctly computed using GPT-2 (8 pts)
- All texts processed and results documented (6 pts)
- Experimental protocol clear and reproducible (6 pts)

3. Statistical Analysis (25 points)

- Descriptive statistics complete (mean, std, min, max) (5 pts)
- 95% confidence intervals correctly calculated (6 pts)
- Independent t-test properly conducted (6 pts)
- Cohen's d effect size calculated and interpreted (5 pts)
- Conclusions drawn from statistical evidence (3 pts)

4. Classification Evaluation (15 points)

- ROC curve correctly generated (5 pts)
- AUC calculated and interpreted (4 pts)
- Optimal threshold determined (3 pts)
- Accuracy, precision, recall reported (3 pts)

5. Error Analysis & Interpretation (15 points)

- Misclassified examples identified and analyzed (5 pts)
- Patterns in false positives/negatives discussed (5 pts)
- Limitations acknowledged (3 pts)
- Practical implications considered (2 pts)

6. Writing Quality & Visualizations (10 points)

- Clear and concise academic writing (3 pts)
- Proper paper structure followed (3 pts)
- Figures clear with error bars where appropriate (2 pts)
- Tables well-formatted and referenced (2 pts)

BONUS POINTS (up to +10)

- Test on different text types (formal vs. casual) (+5)
- Compare multiple AI sources (GPT vs. Claude) (+3)
- Additional statistical tests (e.g., Mann-Whitney U for non-normal data) (+2)

Part 6: Practical Tips

Data Collection Tips

For human texts: - Use the first paragraphs of your paper from Assignment 3 (Self written text) - Copy from public news sources - Keep similar length to AI texts

For AI texts: - Use consistent prompts: “Write a paragraph about [topic]” - Don’t edit or cherry-pick outputs - Match topics to human texts

Good practice:

```
Human: First paragraph from Wikipedia article "Climate Change"
AI: ChatGPT response to "Write a paragraph about climate change"
```

Common Issues

Issue: Perplexity is very high (>1000) - Text might be too short - Special characters causing issues - Try longer texts (50+ words)

Issue: No clear difference between groups - This is a valid finding! Report it honestly - Discuss why detection might be hard

Issue: Code errors with tokenization - Make sure text is clean (no special characters) - Check text isn’t too long (GPT-2 max: 1024 tokens)

Code Template

```python
import torch
import numpy as np
import matplotlib.pyplot as plt
from transformers import GPT2LMHeadModel, GPT2Tokenizer
from sklearn.metrics import roc_curve, auc

# Setup
model = GPT2LMHeadModel.from_pretrained('gpt2')
tokenizer = GPT2Tokenizer.from_pretrained('gpt2')
model.eval()

def calculate_perplexity(text):
    encodings = tokenizer(text, return_tensors='pt', truncation=True, max_length=512)
    with torch.no_grad():
        outputs = model(**encodings, labels=encodings['input_ids'])
        return torch.exp(outputs.loss).item()

# Your texts
human_texts = ["...", "...", ...]
ai_texts = ["...", "...", ...]
```

```
# Calculate perplexities
human_ppls = [calculate_perplexity(t) for t in human_texts]
ai_ppls = [calculate_perplexity(t) for t in ai_texts]

# Analysis...
```

Part 7: Timeline (1.5 weeks)

Days 1-2: Setup & Data

☐ Read DetectGPT abstract
☐ Set up code environment
☐ Collect 20-30 human texts
☐ Generate 20-30 AI texts

Days 3-4: Experiments

☐ Compute all perplexities
☐ Create distribution comparison (Table 1, Figure 1)
☐ Calculate ROC curve (Table 2, Figure 2)

Days 5-6: Analysis & Writing

☐ Error analysis (Experiment 3)
☐ Write Methods and Results sections

Days 7-8: Finish Paper

☐ Write Introduction, Discussion, Conclusion
☐ Create abstract
☐ Proofread and format

Days 9-10: Polish & Submit

☐ Final review
☐ Check all figures are clear
☐ Submit!

Part 8: FAQ

Q: Can I use ChatGPT to generate AND evaluate? A: For generation, yes. For perplexity calculation, use GPT-2 (you need access to log probabilities).

Q: What if there's no clear difference? A: That's a valid scientific finding! Discuss why perplexity alone might not be sufficient.

Q: My perplexity values are all very different from examples online? A: Perplexity values depend on the model. Just compare relative values within your experiment.

Q: Can I use a different model than GPT-2? A: Yes, but GPT-2 is easiest. If using another model, document it clearly. You can use other models as long as your GPU is able to run it. (7b models would be good, to get better results). If you want to use the Univerity GPU Ressources, you can ask us for access.

Q: How do I handle very long texts? A: Truncate to ~500 tokens or use only the first paragraph.

What You'll Learn

After this assignment, you will understand:

1. **Perplexity as a concept** - How LMs measure "surprise"
2. **Practical computation** - Using Hugging Face transformers
3. **AI detection challenges** - Why this is a hard problem
4. **Evaluation metrics** - ROC curves, thresholds, AUC
5. **Scientific thinking** - Hypothesis □ Experiment □ Analysis

This knowledge is valuable for: - Understanding LLM internals - Content moderation systems - Academic integrity tools - General ML evaluation skills