CS224N: Natural Language Processing with Deep Learning

Complete Study Schedule

Based on Stanford CS224N Winter 2021/2023 by Aryan Jain for his own use

12-Week Intensive Learning Plan

Abstract

This comprehensive study plan covers Stanford's CS224N course on Natural Language Processing with Deep Learning. The schedule includes all lecture videos, official assignments, reading materials, and additional practice projects to achieve fluency in NLP. Total estimated time: 12 weeks of dedicated study (15-20 hours/week).

Course Website: https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1214/
Video Playlist: Stanford CS224N YouTube (Winter 2021 & 2023 lectures)

Contents

1 Prerequisites & Setup (Week 0 - Optional)

1.1 Required Knowledge

Before starting, ensure you have:

- Proficiency in Python (NumPy basics)
- Linear Algebra and Calculus fundamentals
- Basic Probability and Statistics
- Machine Learning basics (optional but helpful)

1.2 Environment Setup

1. Install Python 3.8+ with Anaconda/Miniconda
2. Set up PyTorch: https://pytorch.org/
3. Install required libraries: `numpy, matplotlib, jupyter`
4. Create GitHub repository for your assignments
5. Watch: **Python Tutorial (Video 19)** - 47:14
6. Watch: **PyTorch Tutorial (Video 20)** - 47:01
7. Watch: **Hugging Face Tutorial (Video 21)** - 47:57

1.3 Reference Textbooks

- Jurafsky & Martin: *Speech and Language Processing* (free online)
- Goodfellow et al.: *Deep Learning* (free online)
- Goldberg: *A Primer on Neural Network Models for NLP*

2 Week 1: Word Vectors & Foundations

Week 1 Goals

Understand word embeddings, word2vec, and foundational concepts in representing language numerically.

2.1 Day 1-2: Introduction & Word Vectors

Video: Lecture 1 - Intro & Word Vectors (1:24:27)

- Topics: Motivation, word meaning, word2vec, skip-gram model
- Slides: https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1214/ (Jan 12)
- Lecture Notes available on course website

Readings:

- Efficient Estimation of Word Representations in Vector Space (word2vec paper)
- Distributed Representations of Words and Phrases (negative sampling)

Practice:

- Download Gensim word vectors example code
- Experiment with pre-trained word embeddings
- Visualize word similarities using t-SNE

2.2 Day 3-4: Word Vectors 2 & Word Window Classification

Video: Lecture 2 - Neural Classifiers (1:15:18)

- Topics: GloVe, evaluation methods, word window classification
- Slides available (Jan 14)

Readings:

- GloVe: Global Vectors for Word Representation
- Improving Distributional Similarity with Lessons Learned from Word Embeddings

2.3 Day 5-7: Assignment Work

Assignment Due

Assignment 1: Introduction to Word Vectors (6%)
Download from course website, complete all written and coding sections.
Expected time: 8-12 hours

Practice Project

Mini-Project 1: Word Analogy Explorer

- Build tool to explore word analogies (e.g., king - man + woman = queen)
- Test on multiple pre-trained embeddings (Word2Vec, GloVe, FastText)
- Visualize embedding spaces for different domains
- **Time: 4-6 hours**

3 Week 2: Neural Networks & Backpropagation

Week 2 Goals

Master backpropagation, neural network foundations, and implement word2vec from scratch.

3.1 Day 1-2: Backprop & Neural Networks

Video: Lecture 3 - Backprop and Neural Networks (1:22:29)

- Topics: Neural networks, matrix calculus, backpropagation algorithm
- Review CS231n notes on backprop

Readings:

- Matrix calculus notes (course materials)
- CS231n notes on network architectures
- Learning Representations by Backpropagating Errors (Rumelhart et al.)
- "Yes you should understand backprop" (blog post)

Practice:

- Derive gradients for simple neural networks manually
- Implement backprop from scratch (no PyTorch autodiff)
- Verify gradients using numerical gradient checking

3.2 Day 3-7: Assignment Work

Assignment Due

Assignment 2: word2vec Implementation (12%)
Implement word2vec algorithm with derivatives and training loop.
Expected time: 15-20 hours

Key Tasks:

- Derive gradients for skip-gram model
- Implement negative sampling
- Train on small corpus and evaluate
- Analyze learned embeddings

4 Week 3: Dependency Parsing & RNN Foundations

Week 3 Goals

Understand syntactic structure, dependency parsing, and begin sequence modeling with RNNs.

4.1 Day 1-2: Dependency Parsing

Video: Lecture 4 - Syntactic Structure & Dependency Parsing (1:21:22)

- Topics: Constituency vs. dependency, transition-based parsing, neural parsers
- Study annotated slides for detailed explanations

Readings:

- Incrementality in Deterministic Dependency Parsing
- A Fast and Accurate Dependency Parser using Neural Networks
- Universal Dependencies website

4.2 Day 3-4: Recurrent Neural Networks Introduction

Video: Lecture 5 - Recurrent Neural Networks (1:19:18)

- Topics: Language modeling, RNN architecture, backpropagation through time
- Understand why RNNs are needed for sequences

Readings:

- N-gram Language Models (Jurafsky & Martin chapter)
- The Unreasonable Effectiveness of RNNs (Karpathy blog)
- Sequence Modeling chapters from Deep Learning book (10.1, 10.2)

4.3 Day 5-7: Assignment Work

Assignment Due

Assignment 3: Dependency Parsing & Neural Networks (12%)
Build neural dependency parser using PyTorch.
Expected time: 15-20 hours

Practice Project

Mini-Project 2: Character-Level Language Model

- Implement character-level RNN for text generation
- Train on Shakespeare or other literary corpus
- Generate text samples at different temperatures
- Compare vanilla RNN vs. LSTM performance
- **Time: 6-8 hours**

5 Week 4: Advanced RNNs & Sequence-to-Sequence

Week 4 Goals

Master LSTM/GRU architectures, understand vanishing gradients, and learn seq2seq models.

5.1 Day 1-2: LSTMs & Vanishing Gradients

Video: Lecture 6 - Simple and LSTM RNNs (1:21:38)

- Topics: Vanishing/exploding gradients, LSTM, GRU, bidirectional RNNs
- Study LSTM cell diagram carefully

Readings:

- Understanding LSTM Networks (colah's blog - essential!)
- Learning long-term dependencies is difficult (original vanishing gradient paper)
- On the difficulty of training RNNs (proof of vanishing gradient)
- Vanishing Gradients Jupyter Notebook

5.2 Day 3-4: Machine Translation & Seq2Seq

Video: Lecture 7 - Translation, Seq2Seq, Attention (1:18:55)

- Topics: Machine translation, encoder-decoder, attention mechanism
- Understand bottleneck problem in vanilla seq2seq

Readings:

- Sequence to Sequence Learning with Neural Networks (original paper)
- Neural Machine Translation by Jointly Learning to Align and Translate (attention)
- Attention and Augmented RNNs (Distill blog post)
- BLEU metric paper

5.3 Day 5-7: Deep Dive

Practice:

- Implement attention mechanism from scratch
- Visualize attention weights for translation examples
- Compare different attention mechanisms (additive vs. multiplicative)

Additional Reading:

- Statistical Machine Translation (Koehn book chapters)
- Massive Exploration of NMT Architectures (practical advice)

6 Week 5: Neural Machine Translation Assignment

Week 5 Goals

Build complete neural machine translation system with attention and subword modeling.

6.1 Full Week: Assignment 4

Assignment Due

Assignment 4: Neural Machine Translation (12%)
Implement seq2seq with attention for translation task.
Expected time: 20-25 hours (most intensive assignment)

Components:

1. Encoder-decoder architecture
2. Attention mechanism implementation
3. Subword modeling (BPE or WordPiece)
4. Training on translation dataset
5. Evaluation using BLEU score
6. Analysis of attention visualizations

Additional Readings:

- Achieving Open Vocabulary NMT with Hybrid Word-Character Models
- Revisiting Character-Based NMT
- Azure GPU setup guide (if needed for training)

Tips:

- Start early - this is the longest assignment
- Use GPU for training (Google Colab or Azure)
- Debug with small dataset first
- Monitor training curves closely

7 Week 6: Transformers Revolution

Week 6 Goals

Understand transformer architecture, self-attention mechanism, and why transformers replaced RNNs.

7.1 Day 1-3: Transformer Architecture

Video: Lecture 8 - Self-Attention and Transformers (1:17:04)

- Topics: Self-attention, multi-head attention, positional encoding, transformer blocks
- Guest lecture by John Hewitt
- This is one of the most important lectures!

Readings (Critical):

- **Attention Is All You Need** (original transformer paper - read carefully!)
- The Illustrated Transformer (Jay Alammar blog - highly visual)
- Transformer (Google AI blog post)
- Layer Normalization paper

Practice:

- Implement scaled dot-product attention from scratch
- Implement multi-head attention
- Understand positional encoding mathematics
- Compare transformer complexity with RNN

7.2 Day 4-5: Transformers in Practice

Video: Lecture 9 - Pretraining (1:18:46) [2023 version]

- Topics: Pretraining objectives, BERT, GPT, transfer learning
- Guest lecture by John Hewitt

Readings:

- BERT: Pre-training of Deep Bidirectional Transformers
- The Illustrated BERT, ELMo, and co. (Jay Alammar)
- Contextual Word Representations: A Contextual Introduction

7.3 Day 6-7: Extended Applications

Optional Deep Dives:

- Image Transformer paper
- Music Transformer paper
- Apply transformers beyond NLP

Practice Project

Mini-Project 3: Mini-Transformer from Scratch

- Implement simplified transformer for sentiment analysis
- No libraries - build attention, FFN, layer norm manually
- Train on IMDB or similar dataset
- Compare with LSTM baseline
- **Time: 10-12 hours**

8 Week 7: Pretraining & Fine-tuning

Week 7 Goals

Master transfer learning with transformers, understand BERT/GPT paradigms, and implement fine-tuning.

8.1 Day 1-2: Deep Dive on Pretraining

Review Lecture 9 if needed, focus on:

- Masked Language Modeling (MLM)
- Next Sentence Prediction (NSP)
- Causal Language Modeling (CLM)
- Understanding pretraining vs. fine-tuning

8.2 Day 3-7: Assignment Work

Assignment Due

Assignment 5: Self-Supervised Learning & Fine-tuning (12%)
Fine-tune BERT for downstream NLP tasks.
Expected time: 15-20 hours

Tasks:

1. Load pretrained BERT model
2. Fine-tune for text classification
3. Fine-tune for named entity recognition
4. Analyze learned representations
5. Compare different pretrained models

Hugging Face Tutorial:

- Re-watch Video 21 - Hugging Face Tutorial (47:57)
- Practice with `transformers` library
- Explore model hub and datasets

Practice Project

Mini-Project 4: Multi-Task Fine-tuning

- Fine-tune single model for multiple tasks
- Tasks: sentiment, NER, question classification
- Compare multi-task vs. single-task performance
- Implement task-specific heads
- **Time: 8-10 hours**

9 Week 8: Question Answering & Advanced Topics

Week 8 Goals

Understand QA systems, explore advanced architectures, and begin final project planning.

9.1 Day 1-2: Question Answering

Video: Lecture 12 - Question Answering (1:51:53)

- Topics: SQuAD dataset, extractive QA, BiDAF, BERT for QA
- Guest lecture by Danqi Chen
- Longest lecture - take notes carefully

Readings:

- SQuAD: 100,000+ Questions for Machine Comprehension of Text
- Bidirectional Attention Flow for Machine Comprehension
- Reading Wikipedia to Answer Open-Domain Questions
- Dense Passage Retrieval for Open-Domain Question Answering

9.2 Day 3-4: Prompting & RLHF

Video: Lecture 10 - Prompting, RLHF (1:16:15) [2023 version]

- Topics: In-context learning, few-shot prompting, RLHF, instruction tuning
- Modern LLM techniques

Practice:

- Experiment with GPT-3/4 API for few-shot learning
- Design effective prompts for various tasks
- Compare zero-shot vs. few-shot performance

9.3 Day 5-7: Final Project Preparation

Video: Lecture (Practical Tips) - Project Planning

- Review project guidelines
- Choose between Default (SQuAD) or Custom project
- Form team if desired (1-3 people)

Default Project Options:

1. IID SQuAD track - standard QA
2. Robust QA track - out-of-distribution challenges

Custom Project Ideas:

- Multilingual NLP
- Low-resource language processing
- Multimodal learning (text + images)
- Specific domain applications (medical, legal, financial)
- Code generation or program synthesis

10 Week 9: Natural Language Generation & Coreference

Week 9 Goals

Master text generation techniques, understand coreference resolution, and start final project.

10.1 Day 1-2: Natural Language Generation

Video: Lecture 11 - Natural Language Generation (1:18:25) [2023 version]

- Topics: Decoding strategies, beam search, nucleus sampling, evaluation metrics
- Guest lecture by Antoine Bosselut

Readings:

- The Curious Case of Neural Text Degeneration
- Get To The Point: Summarization with Pointer-Generator Networks
- Hierarchical Neural Story Generation
- How NOT To Evaluate Your Dialogue System

Practice:

- Implement different decoding strategies
- Compare greedy, beam search, top-k, nucleus sampling
- Generate text with different temperatures
- Evaluate generation quality (BLEU, ROUGE, perplexity)

10.2 Day 3-4: Coreference Resolution

Video: Lecture 13 - Coreference Resolution (1:21:46)

- Topics: Mention detection, coreference models, neural approaches
- Important for understanding document-level NLP

Readings:

- Coreference Resolution chapter (Jurafsky & Martin)
- End-to-end Neural Coreference Resolution

10.3 Day 5-7: Project Proposal

Assignment Due

Project Proposal Due (5%)
Submit 1-2 page proposal outlining your project.

Proposal Contents:

1. Problem statement and motivation
2. Related work (3-5 papers)
3. Proposed approach
4. Datasets and evaluation metrics
5. Timeline and milestones
6. Team member responsibilities

11 Week 10: Specialized Topics & Project Work

Week 10 Goals

Explore cutting-edge topics, work on project implementation, and deepen practical skills.

11.1 Day 1: Large Language Models

Video: Lecture - T5 and Large Language Models (guest lecture by Colin Raffel)

- Topics: T5 architecture, scaling laws, emergent abilities
- The good, the bad, and the ugly of LLMs

Reading:

- Exploring the Limits of Transfer Learning with T5

11.2 Day 2: Knowledge in Language Models

Video: Lecture 15 - Add Knowledge to Language Models (1:17:25)

- Topics: Knowledge graphs, entity linking, knowledge-enhanced LMs
- Guest lecture by Megan Leszczynski

Readings:

- ERNIE: Enhanced Language Representation with Informative Entities
- Language Models as Knowledge Bases?
- Pretrained Encyclopedia

11.3 Day 3: Multimodal Deep Learning

Video: Lecture 16 - Multimodal Deep Learning (1:18:23) [2023]

- Topics: Vision + language, CLIP, image captioning, VQA
- Guest lecture by Douwe Kiela

11.4 Day 4: Code Generation

Video: Lecture - Code Generation (1:17:57) [2023]

- Topics: Program synthesis, Codex, code understanding
- Modern applications of NLP to programming

11.5 Day 5-7: Project Implementation

Focus on:

- Data collection and preprocessing
- Baseline model implementation
- Initial experiments and debugging
- Setting up evaluation pipeline

12 Week 11: Ethics, Analysis & Project Milestone

Week 11 Goals

Understand ethical considerations, model interpretability, and complete project milestone.

12.1 Day 1: Social & Ethical Considerations

Video: Lecture - Social & Ethical Considerations (guest lecture by Yulia Tsvetkov)

- Topics: Bias in NLP, fairness, privacy, dual use
- Critical lecture for responsible AI

Discussion Topics:

- Bias in word embeddings
- Toxicity in language models
- Privacy concerns with large corpora
- Environmental impact of training

12.2 Day 2: NLP & Linguistics

Video: Lecture 14 - Insights between NLP and Linguistics (1:19:32) [2023]

- Topics: Linguistic structure in neural models, probing tasks
- Bridge between linguistics and deep learning

12.3 Day 3: Model Analysis & Explanation

Video: Lecture 17 - Model Analysis and Explanation (1:17:11)

- Topics: Interpretability, attention analysis, probing classifiers
- Guest lecture by John Hewitt

Video: Lecture - Model Interpretability & Editing (1:11:42) [2023]

- Guest lecture by Been Kim
- Modern interpretability methods

12.4 Day 4-7: Project Milestone

Assignment Due

Project Milestone Due (5%)
Submit progress report with preliminary results.

Milestone Contents:

1. Summary of work completed

2. Preliminary results and analysis
3. Challenges encountered
4. Updated timeline for remaining work
5. Example outputs or visualizations

Practice Project

Mini-Project 5: Model Interpretability Study

- Take your fine-tuned model from Assignment 5
- Implement attention visualization
- Create probing classifiers for linguistic features
- Analyze what the model learned
- **Time: 6-8 hours**

13 Week 12: Future of NLP & Final Project

Week 12 Goals

Complete final project, prepare comprehensive report, and look toward future of NLP.

13.1 Day 1: Future of NLP

Video: Lecture 18 - Future of NLP + Deep Learning (1:20:06)

- Topics: Open problems, research directions, career paths
- Guest lecture by Shikhar Murty
- Inspirational conclusion to the course

Reflect on:

- What you've learned
- Remaining gaps in knowledge
- Areas of interest for deeper study
- Potential research or project ideas

13.2 Day 2-5: Final Project Sprint

Focus Areas:

1. Complete all experiments
2. Run ablation studies
3. Finalize evaluation metrics
4. Create visualizations and tables
5. Compare with baselines and related work

13.3 Day 6-7: Project Report Writing

Assignment Due

Final Project Report Due (30%)
8-10 page report in NeurIPS format.

Report Structure:

1. **Abstract** (150-200 words)
2. **Introduction** (1 page)
   - Problem motivation
   - Research questions
   - Contributions

3. **Related Work** (1-1.5 pages)
    - Survey of relevant papers
    - Position your work
4. **Approach** (2-3 pages)
    - Model architecture
    - Training procedure
    - Hyperparameters
5. **Experiments** (2-3 pages)
    - Dataset description
    - Evaluation metrics
    - Baselines
    - Results tables and figures
6. **Analysis** (1 page)
    - Error analysis
    - Ablation studies
    - Case studies
7. **Conclusion** (0.5 pages)
    - Summary
    - Limitations
    - Future work
8. **References**

Also Submit:

- Project summary image and paragraph (3%)
- Source code (GitHub repository)
- Trained models (if applicable)

14 Additional Practice Projects & Extensions

14.1 Advanced Practice Projects

14.1.1 Project A: Build Your Own Tokenizer

- Implement BPE (Byte Pair Encoding) from scratch
- Implement WordPiece tokenization
- Compare with SentencePiece
- Analyze vocabulary size vs. performance tradeoffs
- **Time: 8-10 hours**

14.1.2 Project B: Sentiment Analysis Pipeline

- Build end-to-end pipeline: data collection → training → deployment
- Scrape Twitter/Reddit data
- Train multiple models (LSTM, BERT, RoBERTa)
- Create REST API for inference
- Build simple web interface
- **Time: 15-20 hours**

14.1.3 Project C: Neural Machine Translation System

- Extend Assignment 4 to full production system
- Implement beam search decoding
- Add ensemble methods
- Create translation API
- Compare transformer vs. RNN+attention
- **Time: 20-25 hours**

14.1.4 Project D: Document Summarization

- Implement extractive summarization (TextRank)
- Implement abstractive summarization (T5/BART)
- Train on CNN/DailyMail dataset
- Evaluate with ROUGE scores
- Build demo application
- **Time: 15-18 hours**

14.1.5 Project E: Named Entity Recognition

- Implement CRF-based NER
- Fine-tune BERT for NER
- Train on CoNLL-2003 dataset
- Create custom entity types
- Build annotation tool
- **Time: 12-15 hours**

14.1.6 Project F: Conversational Chatbot

- Build retrieval-based chatbot
- Implement generative chatbot with GPT
- Create dialogue management system
- Add context tracking
- Deploy on Telegram/Discord
- **Time: 20-25 hours**

15 Weekly Time Commitment & Study Tips

15.1 Expected Time Investment

Activity	Hours/Week
Lecture videos (2-3 per week)	3-5 hours
Reading papers and notes	4-6 hours
Official assignments (when assigned)	10-20 hours
Practice projects	4-8 hours
Review and experimentation	2-4 hours
Total	**15-25 hours**

15.2 Study Strategies

15.2.1 Active Learning

1. **Don't just watch** - implement concepts as you learn
2. **Derive equations** - work through math by hand
3. **Debug thoroughly** - understand why code works
4. **Experiment freely** - try variations on assignments

15.2.2 Paper Reading

1. **First pass**: Read abstract, intro, conclusion (15 min)
2. **Second pass**: Study figures, skim methodology (30 min)
3. **Third pass**: Deep read, take notes, implement (2-3 hours)
4. Keep a reading journal with summaries

15.2.3 Coding Best Practices

1. Write modular, reusable code
2. Document your implementations
3. Use version control (Git)
4. Create unit tests for components
5. Profile code to find bottlenecks

15.2.4 Debugging Deep Learning

1. Start with small data (overfit single batch)
2. Verify data pipeline carefully
3. Check gradient flow (no NaN, reasonable magnitudes)
4. Monitor training curves
5. Use visualization tools (TensorBoard)

16 Resources & Community

16.1 Online Resources

16.1.1 Essential Blogs

- Jay Alammar's blog: https://jalammar.github.io/
- The Gradient: https://thegradient.pub/
- Sebastian Ruder's blog: https://ruder.io/
- Lil'Log (Lilian Weng): https://lilianweng.github.io/
- Distill.pub: https://distill.pub/

16.1.2 Code Repositories

- Hugging Face Transformers: https://github.com/huggingface/transformers
- AllenNLP: https://allennlp.org/
- fairseq (Facebook): https://github.com/pytorch/fairseq
- spaCy: https://spacy.io/

16.1.3 Datasets

- Papers with Code Datasets: https://paperswithcode.com/datasets
- Hugging Face Datasets: https://huggingface.co/datasets
- Common Crawl: https://commoncrawl.org/
- The Pile: https://pile.eleuther.ai/

16.2 Community & Learning

16.2.1 Forums & Discussion

- r/MachineLearning (Reddit)
- r/LanguageTechnology (Reddit)
- Hugging Face Forums
- PyTorch Forums
- Twitter NLP community (#NLProc)

16.2.2 Conferences to Follow

- ACL (Association for Computational Linguistics)
- EMNLP (Empirical Methods in NLP)
- NAACL (North American Chapter of ACL)
- NeurIPS, ICML, ICLR (ML conferences with NLP)

16.2.3 Keeping Up with Research

1. Subscribe to arXiv daily digests (cs.CL, cs.LG)
2. Follow key researchers on Twitter
3. Read Papers with Code trending
4. Watch conference talks on YouTube
5. Join reading groups (virtual or local)

17 Assessment & Grading Breakdown

17.1 Official Course Grading

Component	Percentage	Notes
Assignment 1	6%	Word vectors
Assignment 2	12%	word2vec implementation
Assignment 3	12%	Dependency parsing
Assignment 4	12%	Neural MT
Assignment 5	12%	Transformers & fine-tuning
Assignments Total	**54%**	
Project Proposal	5%	1-2 pages
Project Milestone	5%	Progress report
Project Summary	3%	Image + paragraph
Project Report	30%	8-10 pages
Final Project Total	**43%**	
Participation	3%	Surveys, Ed, guest lectures
Grand Total	**100%**	

17.2 Self-Study Milestones

Since you're self-studying, create your own assessment criteria:

17.2.1 Technical Proficiency

- ☐ Implement attention mechanism from scratch
- ☐ Train transformer model successfully
- ☐ Fine-tune BERT for downstream task
- ☐ Build end-to-end NLP application
- ☐ Debug training issues effectively

17.2.2 Theoretical Understanding

- ☐ Derive backpropagation for RNN
- ☐ Explain attention mechanism intuitively
- ☐ Understand transformer complexity analysis
- ☐ Compare different pretraining objectives
- ☐ Critique NLP research papers

17.2.3 Practical Skills

- ☐ Use Hugging Face ecosystem fluently
- ☐ Design effective prompts for LLMs
- ☐ Evaluate models with appropriate metrics
- ☐ Optimize hyperparameters systematically
- ☐ Deploy model as API/web service

18 Beyond the Course: Next Steps

18.1 Recommended Follow-up Courses

18.1.1 Stanford Courses

- **CS224U**: Natural Language Understanding
- **CS224S**: Spoken Language Processing
- **CS229**: Machine Learning
- **CS231n**: Convolutional Neural Networks for Visual Recognition
- **CS330**: Deep Multi-Task and Meta Learning

18.1.2 Other Online Courses

- **fast.ai**: Practical Deep Learning for Coders
- **DeepLearning.AI**: NLP Specialization
- **Hugging Face**: NLP Course
- **MIT 6.S191**: Introduction to Deep Learning

18.2 Specialization Paths

18.2.1 Path 1: Research in NLP

1. Read 50-100 key papers in your area
2. Replicate important results
3. Identify open problems
4. Propose novel solutions
5. Submit to conferences (ACL, EMNLP, etc.)

18.2.2 Path 2: Applied NLP Engineering

1. Build portfolio of NLP projects
2. Contribute to open-source (Hugging Face, spaCy)
3. Learn MLOps (Docker, Kubernetes, CI/CD)
4. Study production systems design
5. Practice with real-world datasets

18.2.3 Path 3: Specific Domain Expertise

- **Medical NLP**: Clinical notes, drug discovery
- **Legal NLP**: Contract analysis, case law
- **Financial NLP**: Sentiment analysis, earnings calls
- **Social Media NLP**: Hate speech, misinformation
- **Multilingual NLP**: Low-resource languages

18.3 Staying Current

Weekly Routine:

- Browse arXiv (2-3 hours)
- Read 1-2 papers deeply
- Implement interesting ideas
- Discuss with peers/online

Monthly Goals:

- Complete 1 mini-project
- Write blog post about learning
- Contribute to open-source
- Attend virtual meetup

19 Conclusion & Final Tips

19.1 Keys to Success

Golden Rules

1. **Consistency over intensity**: Regular daily study beats cramming
2. **Implementation is key**: Don't just watch - code everything
3. **Understand don't memorize**: Derive equations, explain to others
4. **Debug systematically**: Most learning happens debugging
5. **Build real projects**: Apply knowledge to problems you care about
6. **Join the community**: Learn with others, share your work
7. **Stay curious**: Follow your interests beyond the curriculum
8. **Be patient**: Deep learning takes time to master

19.2 Common Pitfalls to Avoid

- **Passive watching**: Pausing videos to implement is crucial
- **Tutorial hell**: Balance learning with original projects
- **Perfectionism**: Done is better than perfect
- **Isolation**: Engage with community for motivation
- **Skipping fundamentals**: Math and theory matter
- **GPU obsession**: Start with small models on CPU
- **Hype chasing**: Focus on foundations before latest trends

19.3 Motivation

Deep learning for NLP is transforming how we interact with technology. By completing this course, you'll be equipped to:

- Understand how ChatGPT and similar systems work
- Build your own NLP applications
- Contribute to cutting-edge research
- Join a thriving community of practitioners
- Shape the future of human-AI interaction

The journey is challenging but immensely rewarding. Stay persistent, stay curious, and most importantly - enjoy the process of learning!

Good luck with your NLP journey!

Questions? Check course website:
https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1214/

A Quick Reference: Key Equations

A.1 Word2Vec Skip-gram Objective

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\leq j\leq m, j\neq 0} \log P(w_{t+j}|w_t)$$

A.2 Softmax

$$P(w_o|w_c) = \frac{\exp(u_o^T v_c)}{\sum_{w=1}^{V}\exp(u_w^T v_c)}$$

A.3 Negative Sampling

$$\log\sigma(u_o^T v_c) + \sum_{k=1}^{K}\mathbb{E}_{w_k\sim P_n(w)}[\log\sigma(-u_k^T v_c)]$$

A.4 Attention Mechanism

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

A.5 Multi-Head Attention

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1,\ldots,\text{head}_h)W^O$$

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$

A.6 Transformer Feed-Forward

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

A.7 Cross-Entropy Loss

$$L = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

A.8 BLEU Score

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

B Useful Commands & Code Snippets

B.1 PyTorch Setup

```
# Install PyTorch
pip install torch torchvision torchaudio

# Install Transformers
pip install transformers datasets
```

```
# Install utilities
pip install numpy pandas matplotlib seaborn tqdm
```

B.2 Loading Pretrained Models

```
from transformers import AutoModel, AutoTokenizer

model = AutoModel.from_pretrained("bert-base-uncased")
tokenizer = AutoTokenizer.from_pretrained("bert-base-uncased")
```

B.3 Training Loop Template

```
for epoch in range(num_epochs):
    model.train()
    for batch in train_loader:
        optimizer.zero_grad()
        outputs = model(batch)
        loss = criterion(outputs, targets)
        loss.backward()
        optimizer.step()

    model.eval()
    with torch.no_grad():
        val_loss = evaluate(model, val_loader)
```