

Contents

I	Preliminaries	
1	Foundations of Machine Learning	17
1.1	Math Basics	18
1.1.1	Linear Algebra	18
1.1.2	Probability and Statistics	22
1.2	Designing a Text Classifier	26
1.2.1	Problem Statement	27
1.2.2	Documents as Feature Vectors	28
1.2.3	Linear Classifiers	29
1.2.4	Generative vs Discriminative	32
1.2.5	OOV Words and Smoothing	35
1.3	General Problems	37
1.3.1	Supervised and Unsupervised Models	37
1.3.2	Inductive Bias	38
1.3.3	Non-linearity	40
1.3.4	Training and Loss Functions	42
1.3.5	Overfitting and Underfitting	47
1.3.6	Prediction	49
1.4	Model Selection and Evaluation	50
1.4.1	Strategies for Model Selection	51
1.4.2	Training, Validation and Test Data	56
1.4.3	Performance Measure	57
1.4.4	Significance Tests	58
1.5	NLP Tasks as ML Tasks	59
1.5.1	Classification	59
1.5.2	Sequence Labeling	60
1.5.3	Language Modeling/Word Prediction	61
1.5.4	Sequence Generation	62
1.5.5	Tree Generation	63
1.5.6	Relevance Modeling	64

1.5.7	Linguistic Alignment	65
1.5.8	Extraction	67
1.5.9	Others	67
1.6	Summary	68
2	Foundations of Neural Networks	71
2.1	Multi-layer Neural Networks	71
2.1.1	Single-layer Perceptrons	71
2.1.2	Stacking Multiple Layers	73
2.1.3	Computation Graphs	75
2.2	Example: Neural Language Modeling	78
2.3	Basic Model Architectures	83
2.3.1	Recurrent Units	83
2.3.2	Convolutional Units	85
2.3.3	Gate Units	87
2.3.4	Normalization (Standardization) Units	88
2.3.5	Residual Units	89
2.4	Training Neural Networks	90
2.4.1	Gradient Descent	90
2.4.2	Batching	94
2.4.3	Parameter Initialization	96
2.4.4	Learning Rate Scheduling	97
2.5	Regularization Methods	99
2.5.1	Norm-based Penalties	100
2.5.2	Dropout	101
2.5.3	Early Stopping	102
2.5.4	Smoothing Output Probabilities	103
2.5.5	Training with Noise	105
2.6	Unsupervised Methods and Auto-encoders	108
2.6.1	Auto-encoders with Explicit Regularizers	111
2.6.2	Denoising Auto-encoders	113
2.6.3	Variational Auto-encoders	115
2.7	Summary	119

II

Basic Models

3	Words and Word Vectors	123
3.1	Tokenization	124
3.1.1	Tokenization via Rules and Heuristics	125
3.1.2	Tokenization as Language Modeling	126

3.1.3	Tokenization as Sequence Labeling	129
3.1.4	Learning Subwords	130
3.2	Vector Representation for Words	137
3.2.1	One-hot Representation	138
3.2.2	Distributed Representation	138
3.2.3	Compositionality and Contextuality	140
3.3	Count-based Models	142
3.3.1	Co-occurrence Matrices	142
3.3.2	TF-IDF	146
3.3.3	Low-Dimensional Models	147
3.4	Inducing Word Embeddings from NLMs	153
3.5	Word Embedding Models	154
3.5.1	Word2Vec	155
3.5.2	GloVe	157
3.5.3	Remarks	161
3.6	Evaluating Word Embeddings	163
3.6.1	Extrinsic Evaluation	163
3.6.2	Intrinsic Evaluation	164
3.6.3	Visualization	167
3.7	Summary	168
4	Recurrent and Convolutional Sequence Models	171
4.1	Problem Statement	172
4.2	Recurrent Models	173
4.2.1	An RNN-based Language Model	173
4.2.2	Training	175
4.2.3	Layer Stacking	178
4.2.4	Bi-directional Models	180
4.3	Memory	181
4.3.1	Memory as A System	182
4.3.2	Long Short-Term Memory	183
4.3.3	Gated Recurrent Units	185
4.4	Convolutional Models	187
4.4.1	Convolution	187
4.4.2	CNNs for Sequence Modeling	190
4.4.3	Handling Positional Information	193
4.5	Examples	198
4.5.1	Text Classification	198
4.5.2	End-to-End Speech Recognition	200
4.5.3	Sequence Labeling with LSTM and Graphical Models	203

4.5.4	Hybrid Models for Language Modeling	207
4.6	Summary	207
5	Sequence-to-Sequence Models	211
5.1	Sequence-to-Sequence Problems	212
5.2	The Encoder-Decoder Architecture	213
5.2.1	Encoding and Decoding	213
5.2.2	Example: Neural Machine Translation	215
5.3	The Attention Mechanism	218
5.3.1	A Basic Model	219
5.3.2	The QKV Attention	223
5.3.3	Multi-head Attention	226
5.3.4	Hierarchical Attention	229
5.3.5	Multi-layer Attention	232
5.3.6	Remarks	233
5.4	Search	238
5.4.1	The Length Problem	238
5.4.2	Pruning and Beam Search	242
5.4.3	Online Search	250
5.4.4	Exact Search	254
5.4.5	Differentiable Search	256
5.4.6	Hypothesis Diversity	258
5.4.7	Combining Multiple Models	260
5.4.8	More Search Objectives	262
5.5	Summary	265
6	Transformers	269
6.1	The Basic Model	269
6.1.1	The Transformer Architecture	269
6.1.2	Positional Encoding	273
6.1.3	Multi-head Self-attention	274
6.1.4	Layer Normalization	276
6.1.5	Feed-forward Neural Networks	277
6.1.6	Attention Models on the Decoder Side	278
6.1.7	Training and Inference	281
6.2	Syntax-aware Models	283
6.2.1	Syntax-aware Input and Output	284
6.2.2	Syntax-aware Attention Models	285
6.2.3	Multi-branch Models	287
6.2.4	Multi-scale Models	290
6.2.5	Transformers as Syntax Learners	291

6.3	Improved Architectures	295
6.3.1	Locally Attentive Models	295
6.3.2	Deep Models	299
6.3.3	Numerical Method-Inspired Models	305
6.3.4	Wide Models	308
6.4	Efficient Models	312
6.4.1	Sparse Attention	312
6.4.2	Recurrent and Memory Models	317
6.4.3	Low-dimensional Models	322
6.4.4	Parameter and Activation Sharing	327
6.4.5	Alternatives to Self-Attention	328
6.4.6	Conditional Computation	336
6.4.7	Model Transfer and Pruning	341
6.4.8	Sequence Compression	343
6.4.9	High Performance Computing Methods	344
6.5	Applications	347
6.5.1	Language Modeling	348
6.5.2	Text Encoding	349
6.5.3	Speech Translation	350
6.5.4	Vision Models	353
6.5.5	Multimodal Models	355
6.6	Summary	357

III

Large Language Models

7	Pre-training	365
7.1	Pre-training NLP Models	366
7.1.1	Unsupervised, Supervised and Self-supervised Pre-training	366
7.1.2	Adapting Pre-trained Models	368
7.2	Self-supervised Pre-training Tasks	372
7.2.1	Decoder-only Pre-training	372
7.2.2	Encoder-only Pre-training	373
7.2.3	Encoder-Decoder Pre-training	380
7.2.4	Comparison of Pre-training Tasks	386
7.3	Example: BERT	388
7.3.1	The Standard Model	388
7.3.2	More Training and Larger Models	393
7.3.3	More Efficient Models	393
7.3.4	Multi-lingual Models	394

7.4	Applying BERT Models	396
7.5	Summary	401
8	Generative Models	403
8.1	A Brief Introduction to LLMs	404
8.1.1	Decoder-only Transformers	405
8.1.2	Training LLMs	408
8.1.3	Fine-tuning LLMs	409
8.1.4	Aligning LLMs with the World	415
8.1.5	Prompting LLMs	419
8.2	Training at Scale	425
8.2.1	Data Preparation	425
8.2.2	Model Modifications	427
8.2.3	Distributed Training	430
8.2.4	Scaling Laws	433
8.3	Long Sequence Modeling	436
8.3.1	Optimization from HPC Perspectives	437
8.3.2	Efficient Architectures	438
8.3.3	Cache and Memory	441
8.3.4	Sharing across Heads and Layers	450
8.3.5	Position Extrapolation and Interpolation	452
8.3.6	Remarks	463
8.4	Summary	466
9	Prompting	467
9.1	General Prompt Design	468
9.1.1	Basics	468
9.1.2	In-context Learning	471
9.1.3	Prompt Engineering Strategies	473
9.1.4	More Examples	478
9.2	Advanced Prompting Methods	489
9.2.1	Chain of Thought	489
9.2.2	Problem Decomposition	492
9.2.3	Self-refinement	499
9.2.4	Ensembling	505
9.2.5	RAG and Tool Use	509
9.3	Learning to Prompt	515
9.3.1	Prompt Optimization	515
9.3.2	Soft Prompts	519
9.3.3	Prompt Length Reduction	528
9.4	Summary	530

10	Alignment	533
10.1	An Overview of LLM Alignment	534
10.2	Instruction Alignment	535
10.2.1	Supervised Fine-tuning	536
10.2.2	Fine-tuning Data Acquisition	541
10.2.3	Fine-tuning with Less Data	546
10.2.4	Instruction Generalization	547
10.2.5	Using Weak Models to Improve Strong Models	549
10.3	Human Preference Alignment: RLHF	553
10.3.1	Basics of Reinforcement Learning	553
10.3.2	Training Reward Models	560
10.3.3	Training LLMs	563
10.4	Improved Human Preference Alignment	568
10.4.1	Better Reward Modeling	568
10.4.2	Direct Preference Optimization	575
10.4.3	Automatic Preference Data Generation	578
10.4.4	Step-by-step Alignment	580
10.4.5	Inference-time Alignment	583
10.5	Summary	584
11	Inference	587
11.1	Prefilling and Decoding	588
11.1.1	Preliminaries	588
11.1.2	A Two-phase Framework	593
11.1.3	Decoding Algorithms	596
11.1.4	Evaluation Metrics for LLM Inference	607
11.2	Efficient Inference Techniques	608
11.2.1	More Caching	608
11.2.2	Batching	609
11.2.3	Parallelization	619
11.2.4	Remarks	619
11.3	Inference-time Scaling	621
11.3.1	Context Scaling	622
11.3.2	Search Scaling	623
11.3.3	Output Ensembling	623
11.3.4	Generating and Verifying Thinking Paths	624
11.4	Summary	632