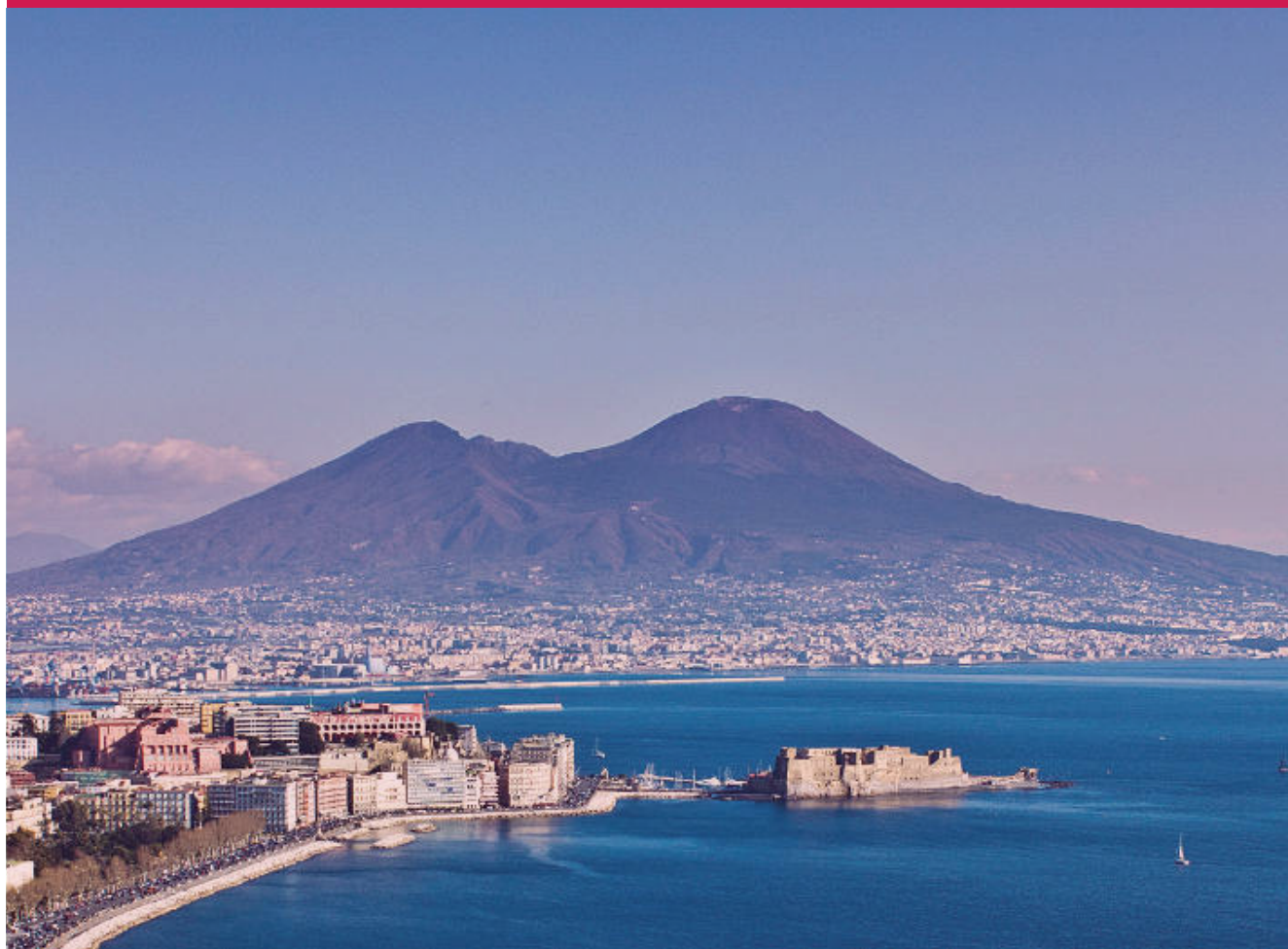


BAYESUVIUS

A VISUAL DICTIONARY OF BAYESIAN
NETWORKS AND CAUSAL INFERENCE



ROBERT R. TUCCI

Bayesuvious,
a visual dictionary of Bayesian Networks and
Causal Inference

Robert R. Tucci
www.ar-tiste.xyz

February 21, 2024

This book is constantly being expanded and improved. To download the latest version, go to <https://github.com/rrtucci/Bayesuvious>

Bayesuvius

by Robert R. Tucci

Copyright ©2020-2023, Robert R. Tucci.

This work is licensed under the Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 United States License. To view a copy of this license, visit the link <https://creativecommons.org/licenses/by-nc-nd/3.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042.



Figure 1: View of Mount Vesuvius from Pompeii



Figure 2: Mount Vesuvius and Bay of Naples

Contents

Foreword	17
Appendices	18
A Navigating the ocean of Judea Pearl's Books	19
B CI-2-3 track	20
C Notational Conventions and Preliminaries	24
C.1 Some abbreviations frequently used throughout this book	24
C.2 Drawing Bayesian Networks	24
C.3 $\mathcal{N}(!a)$	24
C.4 Indicator function (a.k.a. Truth function)	25
C.5 One hot vector	25
C.6 L^p norm	25
C.7 Special sets	26
C.8 Kronecker delta function	26
C.9 Dirac delta function	27
C.10 Majority function	27
C.11 Underlined letters indicate random variables	27
C.12 Probability distributions	28
C.13 Discretization of continuous probability distributions	28
C.14 Samples, i.i.d. variables	29
C.15 Expected Value and Variance	29
C.16 Conditional Expected Value	30
C.17 Notation for covariances	30
C.18 Conditional Covariance	31
C.19 Normal Distribution	32
C.20 Uniform Distribution	34
C.21 Softmax function (a.k.a. Boltzmann Distribution)	34
C.22 Sigmoid and log-odds functions	35
C.23 Estimand, Estimator (curve-fit), Estimate, Bias	36
C.24 Maximum Likelihood Estimate, Likelihood Ratio Test	36
C.25 Mean Square Error (MSE)	37

C.26 Cramer-Rao Bound	39
C.27 Bayes Rule, Bayesian Updating And Conjugate Priors	43
C.28 Linear regression, Ordinary Least Squares (OLS)	45
C.28.1 LR, assuming x_σ are non-random	46
Derivation of LR From Minimization of Error	47
Geometry of LR with non-random x_σ	47
LR Goodness of Fit, R^2	49
C.28.2 LR, assuming x_σ are random	51
Transforming expressions from non-random to random x_σ	52
LR with random x_σ , expressed in derivative notation	54
Double regression of y	58
R^2 with random x_σ	59
C.29 Logistic Regression (LoR)	60
C.30 Entropy, Kullback-Leibler divergence, Cross-Entropy	61
C.31 Definition of various entropies used in Shannon Information Theory	61
C.32 Mean log likelihood asymptotic behavior	63
C.33 Arc Strength (Arc Force)	64
C.34 Pearson Chi-Squared Test	65
C.35 Demystifying Population and Sample Variances	66
C.36 Independence of $\hat{\mu}$ and $\hat{\sigma}^2$	67
C.37 Chi-square distribution	68
C.38 Student's t-distribution	69
C.39 Hypothesis testing and 3 classic test statistics (Likelihood, Score, Wald)	72
C.40 Error Bars	75
C.41 Confidence Interval	76
C.42 Score p-value	79
C.43 Convex/Concave functions, Jensen's Inequality	80
C.44 Chebyshev's inequality	81
C.45 Short Summary of Boolean Algebra	83
C.46 Laplace transform	84
C.46.1 Examples	86
C.46.2 Properties	87
C.47 Z-transform	93
C.47.1 Examples	96
C.47.2 Properties	97
C.48 Legendre Transformation (dual functions)	100
C.48.1 Examples	101
C.48.2 Properties	103
C.48.3 Connection to Fourier transform and Quantum Mechanics	106
C.49 Numpy tensor methods	107

D Definition of a Bayesian Network 113

E	Bayesian Networks, Causality and the Passage of Time	117
E.1	Unifying Principle of this book	117
E.2	You say tomato, I say tomato	118
E.3	A dataset is causal model free	118
E.4	What is causality?	119
E.5	Bayesian Networks and the passage of time	120
E.6	Advice for the DAG-phobic	121
1	AdaBoost	122
1.1	AdaBoost for general ensemble of w-classifiers	122
1.2	AdaBoost for ensemble of tree stumps	126
2	ANOVA	128
2.1	Law of Total Variance	128
2.2	Sum of Squares Estimates	129
2.3	F-statistic and hypothesis testing	131
3	ARACNE structure learning	133
4	Backdoor Adjustment Formula	135
4.1	Examples	136
5	Back Propagation (Automatic Differentiation)	140
5.1	Toy Example	140
5.2	General Theory	141
5.2.1	Jacobians	141
5.2.2	Bnets for function composition, forward propagation and back propagation	142
5.3	Application to Neural Networks	144
5.3.1	Absorbing b_i^λ into $w_{i j}$	144
5.3.2	Bnets for function composition, forward propagation and back propagation for NN	145
5.4	General bnets instead of Markov chains induced by layered structure of NNs	148
6	Bell and Clauser-Horne Inequalities in Quantum Mechanics	149
7	Berkson's Paradox	150
8	Binary Decision Diagrams	152
9	Chow-Liu Trees and Tree Augmented Naive Bayes (TAN)	156
9.1	Chow-Liu Trees	156
9.2	Tree Augmented Naive Bayes (TAN)	160

10 Control Theory (linear, deterministic)	162
10.1 Basic feedback model	163
10.2 Classical model (analog)	164
10.3 Modern model (analog)	167
10.4 Classical model (digital)	171
10.5 Modern model (digital)	171
10.5.1 Discretizing derivatives	172
10.5.2 Solving Difference Equation	173
10.6 Higher than first order differential (or difference) equations	175
10.6.1 Differential Equations	175
10.6.2 Difference Equations	176
10.7 Time-Invariance, Causality, Stability	177
10.8 Controllability, Observability	178
10.9 Signal Flow Graph	178
11 Copula	183
11.1 Examples	186
12 Counterfactual Reasoning	189
12.1 The 3 Rungs of Causal AI	189
12.2 Do operator	190
12.3 Imagine operator	190
13 Cross-Validation	194
14 DAG Extraction From Text (DEFT)	197
15 Dataset Shift and Batch Normalization	198
15.1 Covariate Shift	199
15.2 Concept Shift	199
15.3 Batch Normalization	200
16 Decision Trees	201
16.1 Transforming a dtree into a bnet	203
16.2 Structure Learning for Dtrees	204
16.2.1 Information Gain, Gini	205
16.3 Information Gain Ratio	208
16.3.1 Pseudo-code	209
17 Decisions Based on Rungs 2 and 3: COMING SOON	211

18 Difference-in-Differences	212
18.1 John Snow, DID and a cholera transmission pathway	212
18.2 PO analysis	214
18.3 Linear Regression	216
19 Diffusion Models	219
19.1 Bnet for DM	219
19.2 Mean Values $M^{t-1}(x^t)$ and $M_{\theta}^{t-1}(x^t)$	222
19.3 Loss function \mathcal{L}	225
19.4 Algorithms for training and sampling DM	227
20 Digital Circuits	229
20.1 Mapping any dcircuit to a bnet	229
20.1.1 Option A of Fig.20.2	229
20.1.2 Option B of Fig.20.2	230
21 Do Calculus	231
21.1 3 Rules of Do Calculus	234
21.2 Parent Adjustment Formula	235
21.3 Backdoor Adjustment Formula	237
21.4 Frontdoor Adjustment Formula	238
21.5 Comparison of Backdoor and Frontdoor adjustment formulae	239
21.6 Do operator for DEN diagrams	240
22 Do Calculus proofs	243
23 D-Separation	261
24 D-Separation in Quantum Mechanics	264
25 Dynamical Bayesian Networks	265
26 Expectation Maximization	267
26.1 The EM algorithm:	268
26.1.1 Motivation	269
26.2 Minorize-Maximize (MM) algorithms	269
26.3 Examples	271
26.3.1 Gaussian mixture	271
26.3.2 Blood Genotypes and Phenotypes	272
26.3.3 Missing Data/Imputation	274
27 Factor Graphs	275

28 Frisch-Waugh-Lovell (FWL) theorem	278
28.1 FWL, assuming x^σ are non-random	278
28.2 FWL, assuming x^σ are random	279
29 Frontdoor Adjustment Formula	281
29.1 Examples	282
30 G-formula (Sequential Backdoor Adjustment Formula)	283
31 Gaussian Nodes with Linear Dependence on Parents	287
32 Generalized Linear Model (GLM)	290
32.1 Exponential Family of Distributions	290
32.2 GLM	292
33 Generative Adversarial Networks (GANs)	297
34 Goodness of Causal Fit	302
35 Gradient Descent	303
36 Granger Causality	305
37 Hidden Markov Model	308
37.1 Calculating $P(x_t, v^n)$ and $P(x_t, x_{t+1}, v^n)$	310
37.2 Calculating \mathcal{F}_t and $\overline{\mathcal{F}}_t$	311
37.3 Calculating $P(x^n v^n)$	312
37.4 Calculating $P(v^n A, B, \pi)$	313
37.5 Calculating \hat{x}^n (Viterbi algorithm)	314
37.6 Calculating $\hat{A}, \hat{B}, \hat{\pi}$ (Baum-Welch algorithm)	316
38 Identification of do queries via LDEN diagrams	318
39 Influence Diagrams & Utility Nodes	320
40 Instrumental Inequality and beyond	322
40.1 I-inequality	322
40.1.1 I-inequality for binary z,d,y	324
40.2 Bounds on Effect of IV on treatment outcome y	325
41 Instrumental Variables	328
41.1 δ with unmeasured confounder	328
41.2 δ (with unmeasured confounder) can be inferred via IV	329
41.3 More general bnets with IVs	330
41.4 Instrumental Inequality	331

42 Jackknife Resampling	332
42.1 Case $A = A^n(\vec{x}) = \frac{1}{n} \sum_{\sigma} x^{\sigma}$	334
43 Junction Tree Algorithm	336
44 Kalman Filter	337
44.1 Prediction Problem	338
44.2 Solution	339
44.3 Simple Example	340
44.4 Invariants	341
44.5 Derivation of Solution	341
45 LATE (Local Average Treatment Effect)	343
46 LDEN with feedback loops	349
47 Linear and Logistic Regression	356
47.1 Generalization to x with multiple components (features)	358
47.2 Alternative $V(b, m)$ for logistic regression	358
48 Linear Deterministic Bnets with External Noise	360
48.1 Example of LDEN diagram	360
48.2 LDEN equations and their 2 solutions	361
48.3 Fully connected LDEN diagrams	362
48.3.1 Fully connected LDEN diagram with $nx = 2$	362
48.3.2 Fully connected LDEN diagram with $nx = 3$	364
48.3.3 Fully connected LDEN diagram with arbitrary nx	366
48.4 Not fully connected LDEN diagrams	368
48.5 LDEN diagram with conditioned nodes	369
48.6 SCuMpy	369
48.7 Non-linear DEN diagrams	369
49 Marginalizer Nodes	371
50 Markov Blankets	373
51 Markov Chain Monte Carlo (MCMC)	375
51.1 Inverse Cumulative Sampling	375
51.2 Rejection Sampling	377
51.3 Metropolis-Hastings Sampling	378
51.4 Gibbs Sampling	381
51.5 Importance Sampling	382
52 Markov Chains	384

53 Mediation Analysis	385
54 Mendelian Randomization	392
55 Message Passing and Bethe Free Energy	394
55.1 2MRFs	394
55.2 Message Passing Intuition	395
55.3 $-\ln Z_\theta =$ Free Energy (FE)	399
55.4 $-\ln Z_{\theta^*} =$ Minimum FE	400
55.5 $-\ln Z_\theta^{tree} =$ Tree FE (a.k.a. Bethe FE)	401
55.6 $-\ln Z_{\theta^*}^{tree} =$ Tree Minimum FE, and message passing	402
56 Message Passing, Pearl's theory	406
56.1 Distributed Soldier Counting	406
56.2 Spring Systems	408
56.3 BP for Markov Chains	408
56.4 BP Algorithm for Polytrees	416
56.4.1 How BP algo for polytrees reduces to the BP algo for Markov chains	419
56.5 Derivation of BP Algorithm for Polytrees	420
56.6 Example of BP algo for a Tree	423
56.7 Bipartite bnets	427
56.8 BP for bipartite bnets (BP-BB)	428
56.8.1 BP-BB and general BP agree on Markov chains	430
56.8.2 BP-BB and general BP agree on tree bnets.	432
56.9 BP-BB and sum-product decomposition	434
57 Message Passing in Quantum Mechanics	435
58 Meta-learners for estimating ATE	436
59 Missing Data, Imputation	440
59.1 Imputation via EM	441
59.2 Imputation via MCMC	444
59.3 Multiple Imputations	445
60 Modified Treatment Policy	446
60.1 One time MTP	446
60.2 $\Delta_{ c}$ estimand	450
60.3 Estimates of $\Delta_{ c}$	453
60.3.1 Empirical estimate of $\Delta_{ c}$	453
60.3.2 OR estimate of $\Delta_{ c}$	453
60.4 Other Estimands besides $\Delta_{ c}$	456
60.5 Multi-time MTP	456

61 Monty Hall Problem	459
62 Multi-armed Bandits	461
62.1 Bnet for MAB	462
62.2 Reward functions	464
62.3 Regret functions	466
62.4 Strategies with random exploration	467
62.4.1 ϵ -greedy algorithm	467
62.4.2 ϵ_t -greedy algorithm	468
62.5 Strategies with nonrandom exploration	468
62.5.1 Upper Confidence Bounds (UCB) algorithms	468
Frequentist UCB (UCB1) algorithm	469
Bayesian UCB algorithm	469
62.5.2 Thompson Sampling MAB (TS-MAB) algorithm	471
Bnet for general TS-MAB algorithm	471
TS-MAB algorithm with Beta agent and Bernoulli environment	472
TS-MAB algorithm, skeletal reprise	473
62.5.3 Grad-MAB algorithm	474
63 Naive Bayes	477
64 Neural Networks	478
64.1 Activation Functions $\mathcal{A}_i^\lambda : \mathbb{R} \rightarrow \mathbb{R}$	479
64.2 Weight optimization via supervised training and gradient descent . .	480
64.3 Non-dense layers	482
64.4 Autoencoder NN	484
65 Noisy-OR gate	485
65.1 3 ways to interpret the parameters π_i	486
66 Non-negative Matrix Factorization	490
66.1 Bnet interpretation	490
66.2 Simplest recursive algorithm	491
67 Observationally Equivalent DAGs	492
67.1 Examples	492
68 Omitted Variable Bias	495
69 Personalized Expected Utility	501
69.1 Goal of PEU Theory	502
69.2 Bnets for PEU Theory	503
69.3 Bounds on EU for unspecified bnet	503
69.4 Bounds on EU for specific bnet families	506

70 Personalized Treatment Effects	507
70.1 Goal, Strategy and Rationale of PTE theory	508
70.2 Bnets for PTE theory	510
70.3 $ATE = PB - PH$	511
70.4 Probabilities Relevant to PTE theory	512
70.5 Symmetry	517
70.6 Linear Programming Problem	518
70.7 Special constraints	519
70.8 Matrix representation of probabilities	522
70.9 Bounds on Exp. Probs. imposed by Obs. Probs.	525
70.10 Bounds on $PNS3$ for unspecified bnet	527
70.11 Bounds on $PNS3$ for specific bnet families	533
70.12 Bounds on ATE imposed by Obs. Probs.	533
70.13 Bounds on PNS in terms of ATE and Obs. Probs.	533
70.14 Numerical Examples	534
71 Plate Notation	536
72 Potential Outcomes and Beyond	538
72.1 G and G_{den} bnets, the starting point bnets	539
72.2 G bnet with nodes $y^\sigma(0), y^\sigma(1)$ added to it.	541
72.3 Expected Values of treatment outcome y^σ	543
72.4 Translation Dictionary	543
72.5 $\mathcal{Y}_{ d,x} = \mathcal{Y}_{d d,x}$ (SUTVA)	544
72.6 Conditional Independence Assumption (CIA)	545
72.7 Treatment Effects	545
72.8 Insights into what makes treatment effects equal and $\mathcal{Y}_{1 0} = \mathcal{Y}_1$	548
72.9 G_{do+} bnet	549
72.10 $ACE = ATE$	550
72.11 Good, Bad Controls	551
72.12 PO Confounder Sensitivity Analysis	552
72.13 (SDO, ATE) space	554
72.14 Strata-Matching	557
72.14.1 Exact strata-matching	557
Estimates of Treatment Effects	557
Example, estimation of treatment effects	559
72.14.2 Approximate strata-matching	561
72.14.3 Unbiased strata-matching estimates	561
72.15 Propensities	563
72.16 Propensity based estimates of treatment effects	567
72.17 Positivity	568
72.18 Multi-time PO bnets (Panel Data)	569

73 Program evaluation and review technique (PERT)	573
73.1 Example	575
74 Random Forest and Bagging	579
74.1 Bagging (with fully-featured bags)	579
74.2 Bagging (with randomly-shortened bags)	581
75 Recurrent Neural Networks	582
75.1 Language Sequence Modeling	585
75.2 Other types of RNN	585
75.2.1 Long Short Term Memory (LSTM) unit (1997)	587
75.2.2 Gated Recurrence Unit (GRU) (2014)	589
76 Regression Discontinuity Design	591
76.1 PO analysis	591
76.2 Linear Regression	593
77 Regularization of Loss Functions	594
77.1 L^p norm ROLF	595
77.1.1 L^1 norm ROLF can lead to sparsity	595
77.1.2 L^2 norm ROLF for Least Squares	597
77.2 Proximal functions	598
77.3 Proximal ROLF	600
77.4 Unobserved Nodes of a bnet	601
78 Reinforcement Learning (RL)	603
78.1 Exact RL bnet	606
78.2 Actor-Critic RL bnet	608
78.3 Q function learning RL bnet	610
79 Reliability Box Diagrams and Fault Tree Diagrams	612
79.1 Minimal Cut Sets	618
80 Restricted Boltzmann Machines	620
81 ROC curves	622
81.1 Terminology Table Adapted from Wikipedia Ref.[150]	625
82 Scoring the Nodes of a Learned Bnet	627
82.1 Probability Distributions and Special Functions	628
82.2 Single node with no parents	629
82.3 Multiple nodes with any number of parents	632
82.4 Bayesian Scores	633
82.5 Information Theoretic scores	634

83 Selection Bias Removal	635
83.1 Pre and Post Switch Nodes	636
83.2 Removing SB from passive query $P(y x)$	638
83.3 Removing SB from active query $P(y \mathcal{D}x)$	640
84 Sentence Splitting with SentenceAx	642
84.1 Preliminary Conventions	642
84.1.1 Tensor Notation	642
84.1.2 PyTorch conventions	643
84.2 Bayesian Network for this model	648
84.3 Loss for this model	650
85 Shannon Information Theory	653
86 Shapley Explainability	654
86.0.1 Numerical examples of SHAP	657
87 Simpson's Paradox	660
87.1 Pearl Causality	662
87.2 Numerical Example	664
88 Structure and Parameter Learning for Bnets	665
88.1 Overview	665
88.2 Score based SL algorithms	667
88.3 Constraint based SL algorithms	668
88.4 Pseudo-code for some bnet learning algorithms	669
89 Support Vector Machines And Kernel Method	671
89.1 Learning Algorithm for SVM Classifier	672
89.2 Linear (dot-product) Kernel	673
89.3 Alternatives to Linear Kernel	677
89.4 Random Forest and Kernel Method	677
90 Survival Analysis	678
90.1 $S(t)$ estimates	680
90.1.1 No-censoring estimate of $S(t)$	680
90.1.2 Kaplan-Meier estimate of $S(t)$	680
90.2 $\lambda(t)$ models	685
90.2.1 $\lambda(t)$ independent of covariates Z	685
90.2.2 $\lambda(t)$ dependent on covariates Z	686
90.3 $S_0(t)$ estimates	689
91 Synthetic Controls	691
91.1 PO analysis	693

92 Targeted Estimator	695
92.1 Goal, Strategy, and Rationale of TE theory	695
92.2 Functional Calculus	697
92.3 Linear Approximation of $\Psi[P_N]$	699
92.4 ATE estimand	700
92.5 ATE estimates	701
92.5.1 Ψ^E	701
92.5.2 Ψ^G	702
92.5.3 Ψ^{IPW}	702
92.5.4 Ψ^{LIPW}	702
92.5.5 Ψ^{LIPW++} (a.k.a. Ψ^{TMLE})	706
92.6 Ψ^{TMLE} in practice	709
93 Time Series Analysis: ARMA and VAR	711
93.1 White noise	711
93.2 Backshift operator	712
93.3 Metrics	712
93.4 Definition of $ARMA(p, q)$, $AR(p)$ and $MA(q)$	714
93.5 Solving $AR(p)$	716
93.6 Solving $MA(q)$	717
93.7 Solving $ARMA(p, q)$	718
93.8 Auto-correlation and partial auto-correlation	718
93.9 Generating function of auto-correlation	722
93.10 Impulse Response	723
93.11 $AR(p)$ and Yule-Walker equations	724
93.12 Forecasting	726
93.13 Model Learning	732
93.14 Differencing and $ARIMA(p, d, q)$	732
93.15 Parameter Learning	736
93.15.1 PL of $AR(p)$	736
93.15.2 PL of $MA(q)$	738
93.15.3 PL of $ARMA(p, q)$	741
93.16 $VAR(p)$	742
94 Transfer Learning	743
95 Transformer Networks	745
95.1 Tensor Notation	746
95.2 Recurrent Neural Net with Attention	747
95.2.1 Single Head Attention	747
95.2.2 Multi-Head Attention	750
95.3 Vanilla tranet	752
95.3.1 Single Head Attention	757

95.3.2	Multi-Head Attention	758
95.3.3	Encoder	760
95.3.4	Decoder	762
95.4	BERT	764
95.4.1	BERT parameter values	764
95.4.2	BERT Embedding	765
95.4.3	BERT training	766
96	Transportability of Causal Knowledge	767
97	Turbo Codes	771
97.1	Decoding Algorithm	774
97.2	Message Passing Interpretation of Decoding Algorithm	776
98	Uplift Modelling	777
98.1	UP types	777
98.2	Some Relevant Technical Formulas from Chapter 72	778
98.3	UP Analysis	779
98.4	UP Decision Trees	781
98.4.1	Appendix, connection between Δ_c and Δ_{clj}	786
99	Variational Bayesian Approximation for Medical Diagnosis	787
100	Variational Bayesian Approximation via D_{KL}	791
100.1	Free Energy $\mathcal{F}(\vec{x})$	793
101	XGBoost	796
101.1	Divergences	796
101.2	Minimizing Cost function for single tree	798
101.3	Leaf Splitting	801
101.4	Pruning	801
101.5	Feature Binning	803
101.6	Final estimate of target attribute	803
101.7	Bnet for XGBoost	804
102	YAML for bnet storage	806
102.1	Getting acquainted with YAML	807
102.2	Storing Bnets	808
103	Zero Information Transmission (Graphoid Axioms)	811
103.1	Consequences of Eq.(103.5)	812
	Bibliography	814

Appendices

Chapter 95

Transformer Networks

The primary reference for this chapter is Ref.[81]. Ref.[81] is the highly influential 2017 paper entitled “Attention is all you need” that introduced **Transformer Networks** (tranets) and Attention into the AI vernacular. Besides Ref.[81], I also read blog posts such as Ref.[29] and the Wikipedia article on tranet (Ref. [162]). For a complete list of the large number of excellent blog post that I read to learn about this subject, see my open source software texnn (Ref.[80]).¹

Transformer Networks (tranets) have been taking the fields of Natural Language Processing (NLP) and Large Language Models (LLM) by storm in recent years. They were introduced in 2017 and already are the basis of numerous LLMs. Two famous examples are, BERT (Bidirectional Encoder Representations from Transformers) and ChatGPT (Generative Pre-trained Transformer). Both of these have been trained with huge databases, of which all of the English Wikipedia ($\sim 10^9$ words) is but a small part.

How well ChatGPT works was a huge surprise to most people, including experts in AI/ML. My conjecture is that this surprising LLM performance is due to causality. Let me explain. I believe tranets and the LLM that use them, are just curve-fitters (so are Least Squares, vanilla NNs, Convolutional NNs, etc.). But, we lucked out, because tranets are very good at fitting causal data, and the space of all human generated text, including math equations and computer code, is causally connected (i.e., has a **causally connected topology**).

Normally, tranets are drawn as box diagrams that are somewhat cryptic and ambiguous, at least to me. In this chapter, instead of drawing them as box diagrams, I represent them as causal DAGs (bnets). This makes their causal nature more explicit than the box diagrams, and, in my opinion, also makes them less ambiguous and more understandable than the box diagrams.

Recurrent Neural Nets (RNNs) are discussed in Chapter 75. tranets are quickly displacing RNNs, an older method, in NLP. tranets are better than RNNs for doing

¹texnn is Python software that I wrote specifically for drawing the bnets of this chapter, but later I generalized it to a stand-alone app that can draw any bnet (including SCMs, NNs and tranets), not just a tranet bnet.

NLP in several important ways. Whereas RNNs analyze the tokens (words) of a sentence sequentially (like a Kalman Filter), tranets analyze them in parallel, and thus are more amenable to parallel computing. Also, because RNNs analyze the words of a sentence sequentially, they tend to give more importance to the end of a sentence than to its beginning. That's because RNNs start forgetting the beginning of a sentence by the time they reach its end, like a patient with Alzheimer's. tranets do not suffer from this malady.

Dynamical bnets are discussed in Chapter 25. In Chapter 75, we showed that RNNs are dynamical bnets. In this chapter we will show that tranets are dynamical bnets too.

In this chapter, we will use the Numpy-like tensor notation discussed in Section C.49. In particular, note that $[n] = [0 : n] = \{0, 1, \dots, n - 1\}$ and that $T^{[n],[m]}$ is an $n \times m$ matrix.

95.1 Tensor Notation

Our tensor notation is discussed in Section C.49. Here is a quick review of some of the more salient facts in that section on tensors. Below, we will often accompany an equation in tensor component notation with the equivalent matrix equation, in parenthesis.

We use Greek letters for tensor indices.

Let $\alpha \in [a]$, $\beta \in [b]$, $\gamma \in [c]$, $\delta \in [d]$, $\nu \in [n]$, $\Delta \in [D]$.

- **reshaping**

$$T^{\nu,\delta} \rightarrow T^\Delta \quad (T^{[n_h],[d]} \rightarrow T^{[D]}) \quad (95.1)$$

$$T^\Delta \rightarrow T^{\nu,\delta} \quad (T^{[D]} \rightarrow T^{[n_h],[d]}) \quad (95.2)$$

- **concatenation**

$$T^{[n]} = (T^0, T^1, \dots, T^{n-1}) = (T^\nu)_{\nu \in [n]} \quad (95.3)$$

- **Hadamard product (element-wise, entry-wise multiplication)**

$$T^{[n]} * S^{[n]} = (T^\nu S^\nu)_{\nu \in [n]} \quad (95.4)$$

- **Matrix multiplication**

$T^{[n]} = T^{[n],[1]}$ is a column vector.

$$(T^{[n]})^T S^{[n]} = \text{scalar} \quad (95.5)$$

$$T^{[a],[b]}S^{[b],[c]} = \left[\sum_{\beta \in [b]} T^{\alpha,\beta} S^{\beta,\gamma} \right]_{\alpha \in [a], \gamma \in [c]} \quad (95.6)$$

Most treatments of tranets, including the “Attention is all you need” paper, order the operations chronologically from left to right (L2R). So if A occurs before B , they write AB . This is contrary to what is done in Linear Algebra, where one orders the operations chronologically from right to left (R2L), and one writes BA . In this chapter, will adhere to the Linear Algebra convention, since it is so prevalent and is the overwhelming precedent.

95.2 Recurrent Neural Net with Attention

95.2.1 Single Head Attention

Let

ℓ be the maximum number of words allowed in a sentence. Some words might be blanks (padding).

d be the so called **hidden or embedding dimension**.

$e_\alpha^t \in \mathbb{R}^d$ be a d -dimensional column vector for word $\alpha \in [\ell]$ at time t .

$W_q^t, W_k^t, W_v^t \in \mathbb{R}^{d \times d}$ be the weight matrices for time slice t . The letters Q, K, V stand for Query, Key and Value, respectively. These matrices are learned by training the net. They transform e_α^t as follows

$$v_\alpha^t = W_v^t e_\alpha^t \quad (95.7)$$

$$q_\alpha^t = W_q^t e_\alpha^t \quad (95.8)$$

$$k_\alpha^t = W_k^t e_\alpha^t \quad (95.9)$$

Fig.95.1 represents a tranet of a 3-word sentence as a dynamical bnet. The TPMs (Transition Probability Matrices), printed in blue, for bnet Fig.95.1, are as follows:

$$P(v_\alpha^t | e_\alpha^t) = \mathbb{1}(v_\alpha^t = W_v^t e_\alpha^t) \quad (95.10)$$

$$P(q_\alpha^t | e_\alpha^t) = \mathbb{1}(q_\alpha^t = W_q^t e_\alpha^t) \quad (95.11)$$

$$P(k_\alpha^t | e_\alpha^t) = \mathbb{1}(k_\alpha^t = W_k^t e_\alpha^t) \quad (95.12)$$

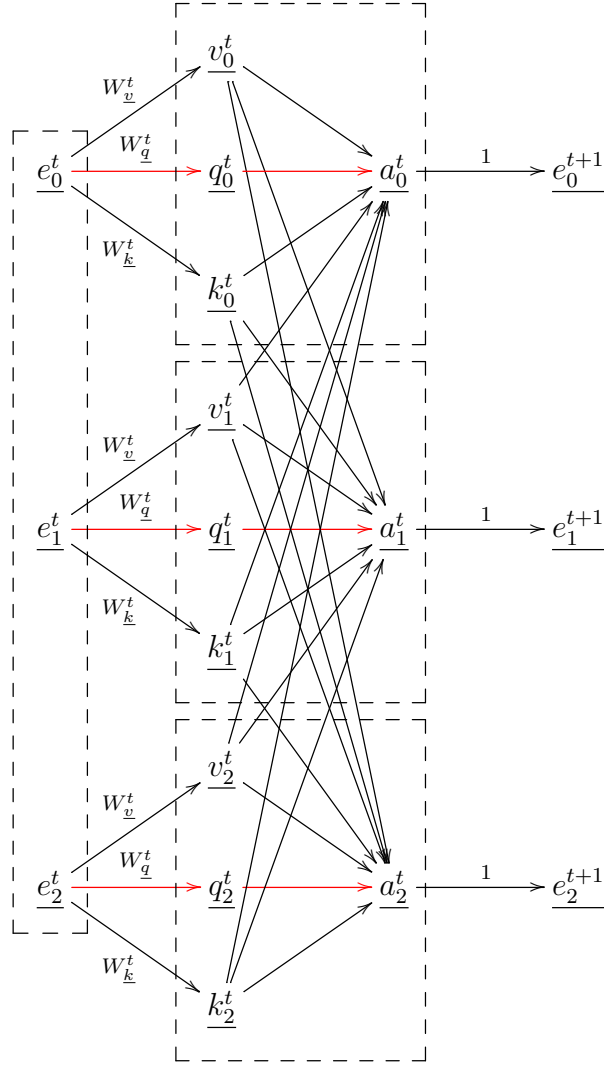


Figure 95.1: Dynamical bnet with single-head Attention for 3 words. Time-slice t . Note that k_α^t for all α points to $a_{\alpha'}^t$ for all α' . Likewise, v_α^t for all α points to $a_{\alpha'}^t$ for all α' . However, q_α^t points only to a_α^t .

$$P(e_\alpha^{t+1}|a_\alpha^t) = \mathbb{1}(e_\alpha^{t+1} = a_\alpha^t) \quad (95.13)$$

$$P(a_\alpha^{t+1}|v_\alpha^t, q_\alpha^t, k_\alpha^t) = \mathbb{1}(a_\alpha^{t+1} = \sum_{\alpha' \in [\ell]} v_{\alpha'}^t P(\alpha'|\alpha)) \quad (95.14)$$

where the conditional probability $P(\alpha'|\alpha)$ is defined as²

²The reason sums over $\delta \in [d]$ are divided by \sqrt{d} is to prevent the argument of the exponential

$$P(\alpha'|\alpha) = \text{softmax} \left[\frac{1}{\sqrt{d}} \sum_{\delta \in [d]} (k^t)^{\delta, [\ell]} (q^t)^{\delta, \alpha} \right] (\alpha'|\alpha) \quad (95.15)$$

$$= \frac{\exp \left(\frac{1}{\sqrt{d}} (k_{\alpha'}^t)^T q_{\alpha}^t \right)}{\sum_{\alpha'' \in [\ell]} \exp \left(\frac{1}{\sqrt{d}} (k_{\alpha''}^t)^T q_{\alpha}^t \right)} \quad (95.16)$$

The right hand side of Eq.(95.14) constitutes an average over all the word vectors $\{v_{\alpha}^t : \alpha \in [\ell]\}$ in a sentence. This average is called the **Attention** (for a single head).³

$$\text{Attention}^{\delta, \alpha} ((v^t)^{[d], [\ell]}, (k^t)^{[d], [\ell]}, (q^t)^{[d], [\ell]}) = \sum_{\alpha' \in [\ell]} (v^t)^{\delta, \alpha'} P(\alpha'|\alpha)$$

(95.17)

On first encounter, the structure of an Attention bnet seems a bit mysterious. Then one realizes that this is an old friend. If the dashed boxes in Fig.95.1 are each “shrunk” to single nodes, then it becomes a TAN Bayes Net. Each of the 3 subgraphs $\underline{e}^t, (\underline{v}^t, \underline{q}^t, \underline{k}^t), \underline{a}^t$ also constitutes a TAN Bayes net.^{4,5} In broad terms, Fig.95.1 can be described by saying that each word undergoes a special kind of 3-class (q,k,v) Naive Bayes classification, and the results of that classification are sent to the new version of every word (except the q class which only sends info to one word, not all of them).

It’s also useful to think of Attention as a filter with input signal $(e^t)^{[d], [\ell]}$ and output signal $(e^{t+1})^{[d], [\ell]}$.

Fig.95.1 can be “folded” (i.e., the 3 words can be represented by as single node). When folded, Fig.95.1 becomes Fig.95.2. Note that in Fig.95.2, we have started indicating the shapes of tensors by a superscript, using the tensor notation explained in Section C.49. We will continue doing this henceforth in this chapter.

The structural equations for Fig.95.2, printed in blue, are as follows.

$$(a^t)^{[d], [\ell]} = \text{Attention}((v^t)^{[d], [\ell]}, (k^t)^{[d], [\ell]}, (q^t)^{[d], [\ell]}) \quad (95.18a)$$

from getting too large.

³Variations of this definition of Attention have been proposed. This particular one is the original one from the “Attention is all you need paper”. Some people call it the “scaled dot product Attention”.

⁴Tree Augmented Naive (TAN) Bayes nets were introduced in Chapter 9.

⁵A **reverse or upside down tree** is obtained by reversing the directions of all the arrows of a tree directed graph. A TAN Bayes net is normally defined as in Chapter9, as a Naive Bayes net augmented with a tree. In an Attention bnet, the Naive Bayes Net is augmented with a reverse tree (RT) instead of a tree (T), so technically Attention bnets contain RTAN Bayes nets, not TAN Bayes nets.

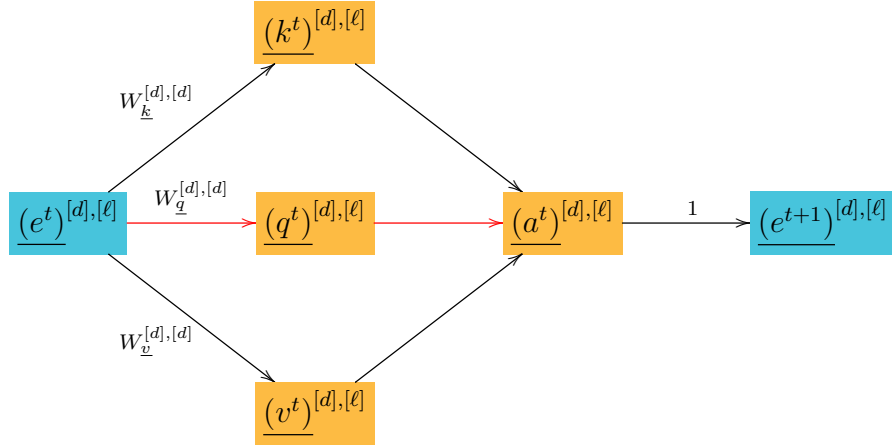


Figure 95.2: Folded version of Fig.95.1 when $\ell = 3$. Note that all orange nodes have the same tensor shape.

$$(e^t)^{[d],[\ell]} = \text{prior} \quad (95.18b)$$

$$(e^{t+1})^{[d],[\ell]} = (a^t)^{[d],[\ell]} \quad (95.18c)$$

$$(k^t)^{[d],[\ell]} = W_{\underline{k}}^{[d],[d]}(e^t)^{[d],[\ell]} \quad (95.18d)$$

$$(q^t)^{[d],[\ell]} = W_{\underline{q}}^{[d],[d]}(e^t)^{[d],[\ell]} \quad (95.18e)$$

$$(v^t)^{[d],[\ell]} = W_{\underline{v}}^{[d],[d]}(e^t)^{[d],[\ell]} \quad (95.18f)$$

95.2.2 Multi-Head Attention

In this section, we will generalize the single head Attention, as defined in the previous section, to multi-head Attention.

Let

n_h = number of heads. $\nu \in [n_h]$.

d = same as before, the hidden, embedding dimension. $\delta \in [d]$

$D = n_{\underline{h}}d$. $\Delta \in [D]$. We will do some tensor reshaping: $T^{[n_{\underline{h}}], [d]} \rightarrow T^{[D]}$, or, in component form, $T^{\nu, \delta} \rightarrow T^{\Delta}$.

Consider weight matrices $W_{\underline{k}}^{[D], [d]}$, $W_{\underline{q}}^{[D], [d]}$, and $W_{\underline{v}}^{[D], [d]}$ such that

$$(k^t)^{\nu, \delta, \alpha} = \sum_{\delta' \in [d]} W_{\underline{k}}^{\nu, \delta, \delta'} (e^t)^{\delta', \alpha} \quad (95.19)$$

$$(q^t)^{\nu, \delta, \alpha} = \sum_{\delta' \in [d]} W_{\underline{q}}^{\nu, \delta, \delta'} (e^t)^{\delta', \alpha} \quad (95.20)$$

$$(v^t)^{\nu, \delta, \alpha} = \sum_{\delta' \in [d]} W_{\underline{v}}^{\nu, \delta, \delta'} (e^t)^{\delta', \alpha} \quad (95.21)$$

We define the **Multi-head Attention** by

$$\text{Attention}^{\nu, \delta, \alpha} \left((v^t)^{[D], [\ell]}, (k^t)^{[D], [\ell]}, (q^t)^{[D], [\ell]} \right) = \sum_{\alpha' \in [\ell]} (v^t)^{\nu, \delta, \alpha'} P(\alpha' | \alpha, \nu) \quad (95.22)$$

where

$$P(\alpha' | \alpha, \nu) = \text{softmax} \left[\frac{1}{\sqrt{d}} \sum_{\delta \in [d]} (k^t)^{\nu, \delta, [\ell]} (q^t)^{\nu, \delta, \alpha} \right] (\alpha' | \alpha, \nu) \quad (95.23)$$

$$= \frac{\exp \left[\frac{1}{\sqrt{d}} \sum_{\delta \in [d]} (k^t)^{\nu, \delta, \alpha'} (q^t)^{\nu, \delta, \alpha} \right]}{\sum_{\alpha'' \in [\ell]} \exp \left[\frac{1}{\sqrt{d}} \sum_{\delta \in [d]} (k^t)^{\nu, \delta, \alpha''} (q^t)^{\nu, \delta, \alpha} \right]} \quad (95.24)$$

The structural equations, printed in blue, for the bnet Fig.95.3, are as follows. Note that Attention() always has the same tensor shape as its 3 arguments. Note also that the 3 weight matrices $W_{\underline{k}}^{[D], [d]}$, $W_{\underline{q}}^{[D], [d]}$, and $W_{\underline{v}}^{[D], [d]}$ raise the hidden dimension, whereas the weight matrix $W_{\underline{a}}^{[d], [D]}$ lowers it. $W_{\underline{a}}^{[d], [D]} = 1$ in the single head case.

$$(a^t)^{[D], [\ell]} = \text{Attention}((v^t)^{[D], [\ell]}, (k^t)^{[D], [\ell]}, (q^t)^{[D], [\ell]}) \quad (95.25a)$$

$$(e^t)^{[d], [\ell]} = \text{prior} \quad (95.25b)$$

$$(e^{t+1})^{[d], [\ell]} = W_{\underline{a}}^{[d], [D]} (a^t)^{[D], [\ell]} \quad (95.25c)$$

$$(k^t)^{[D], [\ell]} = W_{\underline{k}}^{[D], [d]} (e^t)^{[d], [\ell]} \quad (95.25d)$$

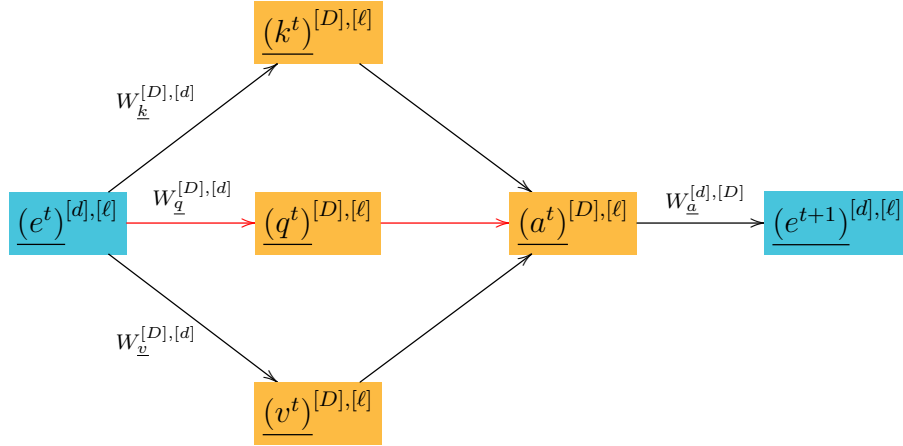


Figure 95.3: Dynamical bnet with single-head Attention for ℓ words. Time-slice t . This is a generalization of the single head Attention of Fig.95.2. Note that all orange nodes have the same tensor shape.

$$(q^t)^{[D],[\ell]} = W_{\underline{q}}^{[D],[d]}(e^t)^{[d],[\ell]} \quad (95.25e)$$

$$(v^t)^{[D],[\ell]} = W_{\underline{v}}^{[D],[d]}(e^t)^{[d],[\ell]} \quad (95.25f)$$

95.3 Vanilla tranet

In this section, we will discuss the tranet of the “Attention is all you need” paper, Ref.[81]. As is common in the literature, we will refer to that tranet as the “Vanilla” tranet. Ref.[81] describes its tranet graphically with Fig.95.4. Our goal is to find a causal DAG (bnet) version of that figure.

Let

ℓ =, context window, maximum number of words in a sentence segment. $\alpha \in [\ell]$, $\ell \sim 100$

L = number of words in vocabulary, $\beta \in [L]$, $L \gg \ell$

$d = d_{\underline{q}} = d_{\underline{k}} = d_{\underline{v}} = 64$, hidden dimension per head, $\delta \in [d]$.

$n_{\underline{h}} = 8$, number of heads, $\nu \in [n_{\underline{h}}]$

$D = n_{\underline{h}}d = 8(64) = 512$, hidden dimension for all heads, $\Delta \in [D]$

$\Lambda = 6$, number copies, connected in series, of boxed bnet, $\lambda \in [\Lambda]$

Before we present the bnet version of Fig.95.4, we discuss some of the definitions needed to understand and motivate Fig.95.4.

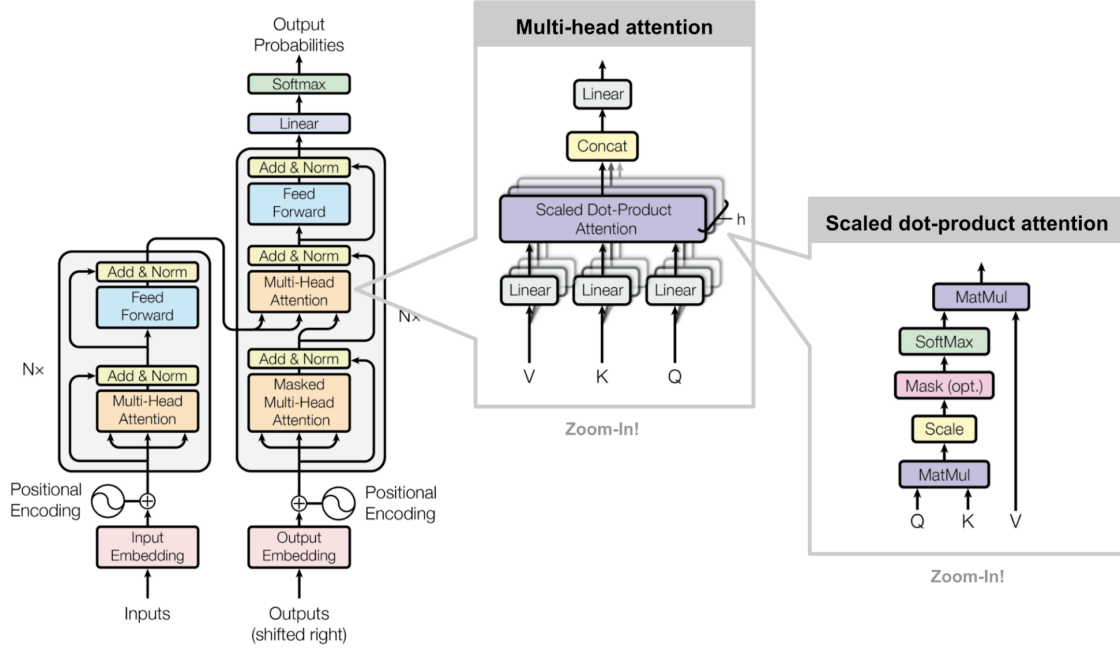


Figure 95.4: Vanilla tranet

- **Encoder Input** $x^{\beta, \alpha}$

$$x^{\beta, \alpha} = \delta(\beta, \beta(\alpha)) \left(x^{[L], [\ell]} \text{ has one hot columns.} \right) \quad (95.26)$$

- **Embedding (a.k.a. encoding) Matrix** $\mathcal{E}^{\delta, \beta}$

$$e^{\delta, \alpha} = \sum_{\beta} \mathcal{E}^{\delta, \beta} x^{\beta, \alpha} \quad (e^{[d], [\ell]} = \mathcal{E}^{[d], [L]} x^{[L], [\ell]}) \quad (95.27)$$

- **Weight matrices** $W_{\underline{q}}, W_{\underline{k}}, W_{\underline{v}}$

$$Q^{\nu, \delta, \alpha} = \sum_{\delta'} W_{\underline{q}}^{\nu, \delta, \delta'} e^{\delta', \alpha} \quad (Q^{[D], [\ell]} = W_{\underline{q}}^{[D], [d]} e^{[d], [\ell]}) \quad (95.28)$$

$$K^{\nu, \delta, \alpha} = \sum_{\delta'} W_{\underline{k}}^{\nu, \delta, \delta'} e^{\delta', \alpha} \quad (K^{[D], [\ell]} = W_{\underline{k}}^{[D], [d]} e^{[d], [\ell]}) \quad (95.29)$$

$$V^{\nu, \delta, \alpha} = \sum_{\delta'} W_{\underline{v}}^{\nu, \delta, \delta'} e^{\delta', \alpha} \quad (V^{[D], [\ell]} = W_{\underline{v}}^{[D], [d]} e^{[d], [\ell]}) \quad (95.30)$$

- **Multi-head Attention**

$$B^{\nu,\alpha',\alpha} = \frac{1}{\sqrt{d}} \sum_{\delta} K^{\nu,\delta,\alpha'} Q^{\nu,\delta,\alpha} \quad \left(B^{\nu,\alpha',\alpha} = \frac{1}{\sqrt{d}} (K^{\nu,[d],\alpha'})^T Q^{\nu,[d],\alpha} \right) \quad (95.31)$$

$$A^{\nu,\delta,\alpha} = \sum_{\alpha'} V^{\nu,\delta,\alpha'} \underbrace{\text{softmax}(B^{\nu,[\ell],\alpha})(\alpha'|\alpha, \nu)}_{P(\alpha'|\alpha, \nu)} \quad (95.32)$$

$$\sum_{\alpha' \in [\ell]} P(\alpha'|\alpha, \nu) = 1 \quad (95.33)$$

$$A^{\nu,\delta,\alpha} \rightarrow A^{\Delta,\alpha} (A^{[n_h],[d],[\ell]} \rightarrow A^{[D],[\ell]}) \quad (95.34)$$

Important: Note that the $\text{softmax}()$ makes the α' component a probability, not the α one!

For example, suppose $\nu = 1$ (one head), $\ell = 2$ (a 2 word segment), and $d = 3$ (hidden dimension is 3). The $Q^{[3],[2]}, K^{[3],[2]}, V^{[3],[2]}$ are 3×2 matrices (i.e., two 3-dim column vectors). One uses the $Q^{[3],[2]}$ and $K^{[3],[2]}$ to arrive at a 2×2 matrix $P(\alpha'|\alpha)$ of probabilities. Then one uses that matrix of probabilities to replace

$$[V^{[3],0}, V^{[3],1}] \rightarrow [V^{[3],0}P(0|0) + V^{[3],1}P(1|0), V^{[3],0}P(0|1) + V^{[3],1}P(1|1)] \quad (95.35)$$

- **Positional Embedding Matrix $\mathcal{E}_{pos}^{\delta,\beta}$**

$$\mathcal{E}_{pos}^{\delta,\beta} = \begin{cases} \sin\left(2\pi \frac{\beta}{(2\pi)10^{4\delta/d}}\right) = \sin\left(2\pi \frac{\beta}{\lambda(\delta)}\right) & \text{if } \delta \text{ is even} \\ \cos\left(2\pi \frac{\beta}{(2\pi)10^{4(\delta-1)/d}}\right) = \cos\left(2\pi \frac{\beta}{\lambda(\delta)}\right) & \text{if } \delta \text{ is odd} \end{cases} \quad (95.36)$$

$\mathcal{E}_{pos}^{\delta,\beta}$ changes in phase by $\pi/2$ every time δ changes by 1. Its wavelength λ is independent of β , but increases rapidly with δ , from $\lambda(\delta = 0) = 2\pi * 1$ to $\lambda(\delta = d) = 2\pi * 10^4$.

Total Embedding equals initial embedding plus positional embedding:

$$\mathcal{E}^{\delta,\beta} = \mathcal{E}_0^{\delta,\beta} + \mathcal{E}_{pos}^{\delta,\beta} \quad (95.37)$$

The purpose of positional embedding is to take $e^{\beta,\alpha}$ to $e^{\delta,\alpha} = \sum_{\beta} \mathcal{E}_{pos}^{\delta,\beta} e^{\beta,\alpha}$ where $e^{\delta,\alpha}$ changes quickly as δ (i.e., position) changes.

- **ReLU**

For a tensor T of arbitrary shape,

$$ReLU(T) = (T)_+ = \max(0, T) \quad (95.38)$$

max element-wise.

- **Feed Forward Neural Net**

$$F(e^{\delta, \alpha}) = \sum_{\Delta \in [n_{ff}]} W_2^{\delta, \Delta} ReLU \left(\sum_{\delta' \in [d]} W_1^{\Delta, \delta'} e^{\delta', \alpha} + b_1^{\Delta, \alpha} \right) + b_2^{\delta, \alpha} \quad (95.39)$$

n_{ff} is called the `intermediate_size` in BERT.

- **Softmax**

`softmax()` takes a vector and returns a vector of probabilities of the same length

$$e^{[n]} \rightarrow P^{[n]} \quad (95.40)$$

where

$$P^\alpha = \frac{\exp(e^\alpha)}{\sum_{\alpha \in [n]} \exp(e^\alpha)} \left(P^{[n]} = \frac{\exp(e^{[n]})}{\| \exp(e^{[n]}) \|_0} \right) \quad (95.41)$$

For example,

$$(1, 0, 0) \rightarrow (e, 1, 1)/norm \quad (95.42)$$

$$(10, 0, 0) \rightarrow (e^{10}, 1, 1)/norm \approx (1, 0, 0) \quad (95.43)$$

For any $a \in \mathbb{R}$,

$$(a, a, a) \rightarrow \frac{1}{3}(1, 1, 1) \quad (95.44)$$

- **Skip Connection (Add & Normalize)**

A **skip connection** is when you split the input to a **filter** into two streams, one stream goes through the filter, the other doesn't. The one that doesn't is then merged with the output of the filter via a **add & normalize** node. The reason for making skip connections is that the signal exiting a filter is usually full of jumps and kinks. By merging that filter output with some of the filter input, one smooths out the filter output to some degree. This makes back-propagation differentiation better behaved.

The filter might be a Multi-Head Attention or a Feed Forward NN.

Add & Normalize just means $(A + B)/norm$ where A and B are the two input signals and “norm” is some norm of $A + B$ (for instance, $\|A + B\|_2$).

Normalization keeps the signal from growing too big and saturating the signal that will enter components upstream. Normalization can also involve subtracting the mean $\langle X \rangle$ of the signal X so as to get a signal $X - \langle X \rangle$ that has zero mean.

- **Redundancy**

For better results, the Encoder and Decoder both contain Λ copies, connected in series, of the boxed bnet.

Redundancy (see Chapter 79) has been used to avoid catastrophic failure at least as early as the dawn of the age of rocketry, when it was used to avoid the all too common occurrence of exploding rockets. There are 2 basic types of redundancy: in series connection (as in the repeated identical layers in a feedforward NN or a recurrent NN), and in parallel connection (as in tranet heads, and the plates in a bnet (see Chapter 71)).

- **Right Shifted Outputs**

“Outputs (Shifted Right)” in Fig.95.4 refers to what is called **forced teaching** in the RNN (recurrent neural net) literature. We explain forced teaching in Fig.95.5.

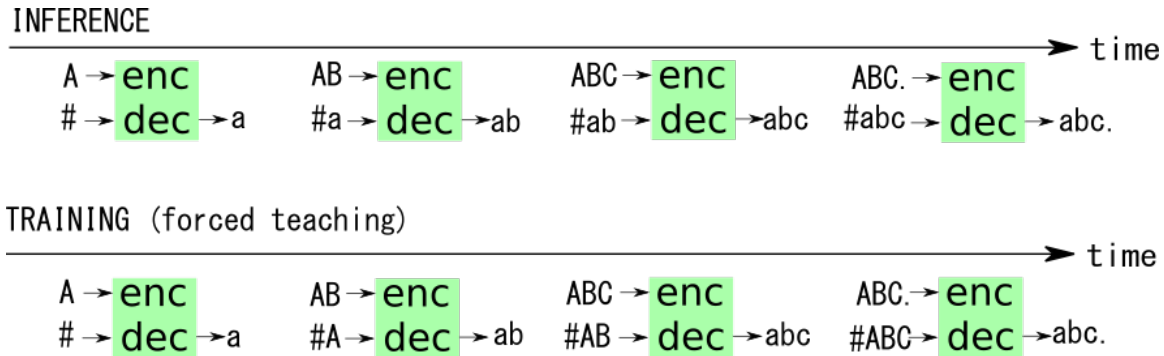


Figure 95.5: Training and Inference for vanilla transformer. “enc” and “dec” denote the encoder and decoder, respectively. A hash character represents the SOS (start of sentence) token, and a period represents the EOS (end of sentence) token. Capital letters represent ground truth tokens, and lower case ones represent predictions.

- **Masked Attention**

$$P(\alpha'|\alpha, \nu) = 0 \quad \text{if } \alpha' < \alpha \quad (95.45)$$

α , and α' are word positions in a sentence, and α' is in the future (downstream) compared to α . So as to not violate causality, this condition enforces the constraint that no attention is paid to word positions in the future of α .

95.3.1 Single Head Attention

Fig.95.6 gives a bnet representation of the “Single Head Attention” portion of Fig.95.4. The structural equations for that bnet, printed in blue, are as follows.

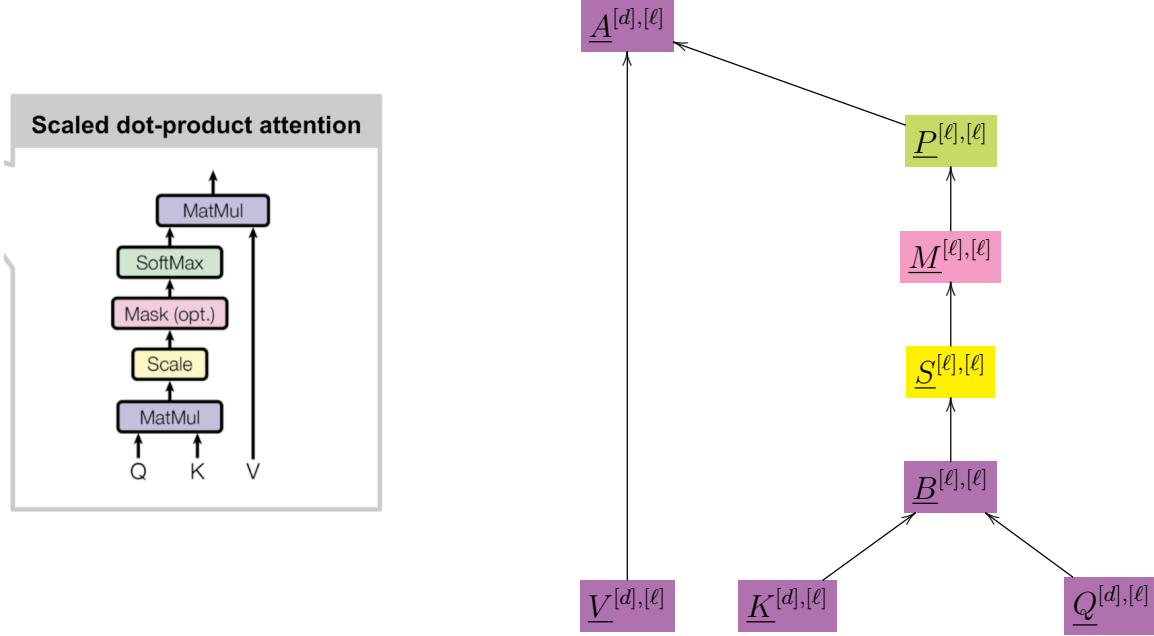


Figure 95.6: Single Head Attention. (Scaled Dot Product)

$$A^{[d],[\ell]} = V^{[d],[\ell]} P^{[\ell],[\ell]} \quad \left(\text{Note that } \sum_{\alpha \in [\ell]} P^{\alpha,[\ell]} = 1 \right) \quad (95.46a)$$

$$B^{[\ell],[\ell]} = (K^{[d],[\ell]})^T Q^{[d],[\ell]} \quad (95.46b)$$

$$K^{[d],[\ell]} = \text{prior} \quad (95.46c)$$

$$M^{[\ell],[\ell]} = \text{mask}(S^{[\ell],[\ell]}) \quad (95.46d)$$

$$P^{[\ell],[\ell]} = \text{softmax}(M^{[\ell],[\ell]}) \quad \left(\text{Note that } \sum_{\alpha \in [\ell]} P^{\alpha,[\ell]} = 1 \right) \quad (95.46e)$$

$$Q^{[d],[\ell]} = \text{prior} \quad (95.46f)$$

$$S^{[\ell],[\ell]} = \frac{B^{[\ell],[\ell]}}{\sqrt{d}} \quad (95.46g)$$

$$V^{[d],[\ell]} = \text{prior} \quad (95.46h)$$

95.3.2 Multi-Head Attention

Fig.95.7 gives a bnet representation of the “Multi-Head Attention” portion of Fig.95.4. The structural equations for that bnet, printed in blue, are as follows.

$$A^{[D],[\ell]} = [A_0^{[d],[\ell]} | A_1^{[d],[\ell]}] \quad (95.47a)$$

$$A_0^{[d],[\ell]} = \text{Attention}(V_0^{[d],[\ell]}, K_0^{[d],[\ell]}, Q_0^{[d],[\ell]}) \quad (95.47b)$$

$$A_1^{[d],[\ell]} = \text{Attention}(V_1^{[d],[\ell]}, K_1^{[d],[\ell]}, Q_1^{[d],[\ell]}) \quad (95.47c)$$

$$K^{[D],[\ell]} = W_{\underline{k}}^{[D],[d]} e^{[d],[\ell]} \quad (95.47d)$$

$$K_0^{[d],[\ell]} = \text{linear}(K^{[D],[\ell]}) \quad (\text{split, then project a component}) \quad (95.47e)$$

$$K_1^{[d],[\ell]} = \text{linear}(K^{[D],[\ell]}) \quad (\text{split, then project a component}) \quad (95.47f)$$

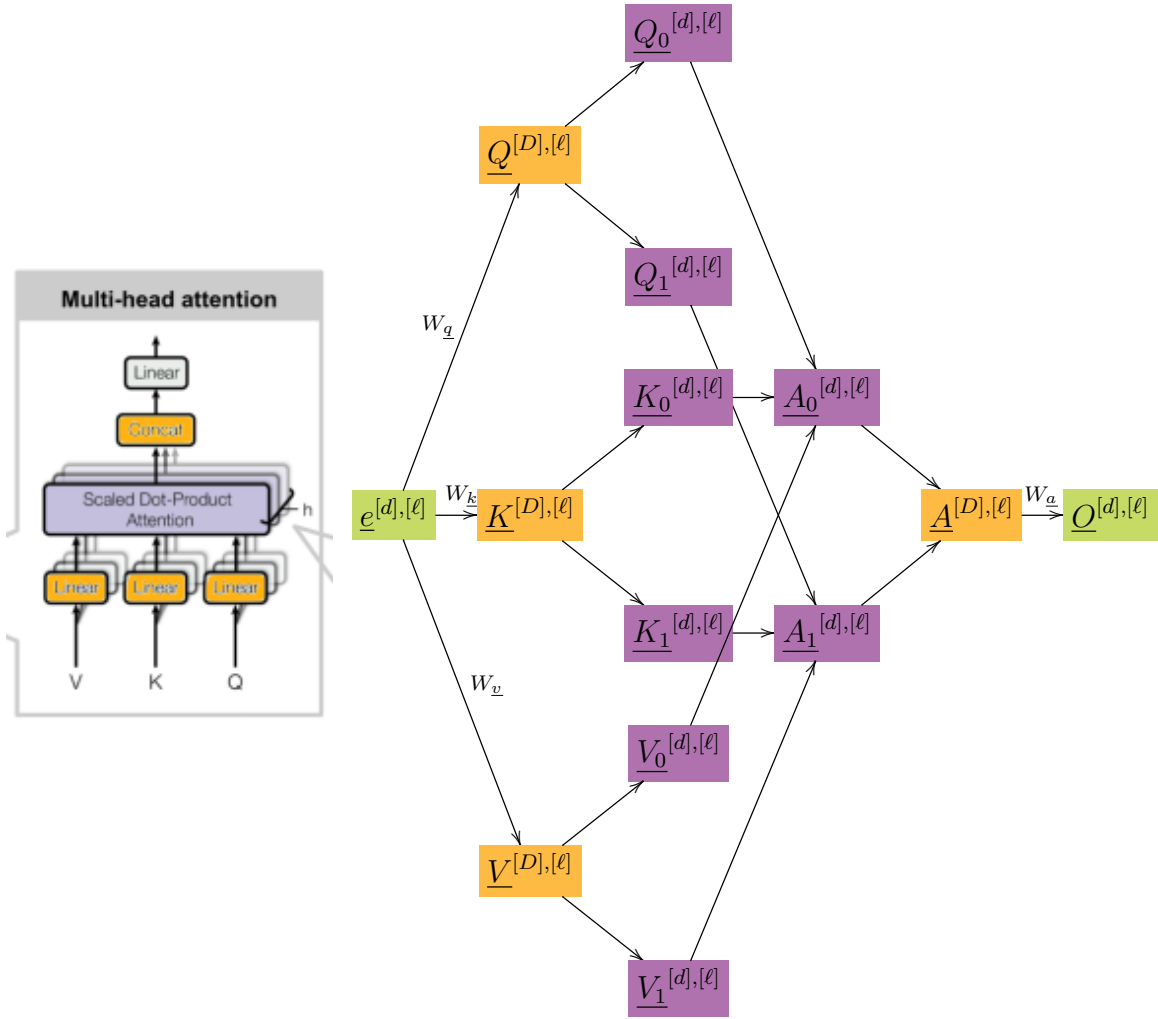


Figure 95.7: Multi-head Attention with 2 heads. Note that the orange nodes all have the same tensor shape.

$$O^{[d],[\ell]} = W_{\underline{a}}^{[d],[D]} A^{[D],[\ell]} \quad (95.47g)$$

$$Q^{[D],[\ell]} = W_{\underline{q}}^{[D],[d]} e^{[d],[\ell]} \quad (95.47h)$$

$$Q_0^{[d],[\ell]} = \text{linear}(Q^{[D],[\ell]}) \quad (\text{split, then project a component}) \quad (95.47i)$$

$$Q_1^{[d],[\ell]} = \text{linear}(Q^{[D],[\ell]}) \quad (\text{split, then project a component}) \quad (95.47j)$$

$$V^{[D],[\ell]} = W_{\underline{v}}^{[D],[d]} e^{[d],[\ell]} \quad (95.47k)$$

$$V_0^{[d],[\ell]} = \text{linear}(V^{[D],[\ell]}) \quad (\text{split, then project a component}) \quad (95.47l)$$

$$V_1^{[d],[\ell]} = \text{linear}(V^{[D],[\ell]}) \quad (\text{split, then project a component}) \quad (95.47m)$$

$$e^{[d],[\ell]} = \text{prior} \quad (95.47n)$$

95.3.3 Encoder

Fig.95.8 gives a bnet representation of the "Encoder" portion of Fig.95.4. The structural equations for that bnet, printed in blue, are as follows.

$$A^{[D],[\ell]} = \text{Attention}(Q^{[D],[\ell]}, K^{[D],[\ell]}, V^{[D],[\ell]}) \quad (95.48a)$$

$$e^{[d],[\ell]} = \mathcal{E}^{[d],[L]} x^{[L],[\ell]} \quad (95.48b)$$

$$F^{[d],[\ell]} = \text{feed_forward_nn}(N^{[d],[\ell]}) \quad (95.48c)$$

$$K^{[D],[\ell]} = W_{\underline{k}}^{[D],[d]} e^{[d],[\ell]} \quad (95.48d)$$

$$n^{[d],[\ell]} = \text{normalize}(N^{[d],[\ell]} + F^{[d],[\ell]}) \quad (95.48e)$$

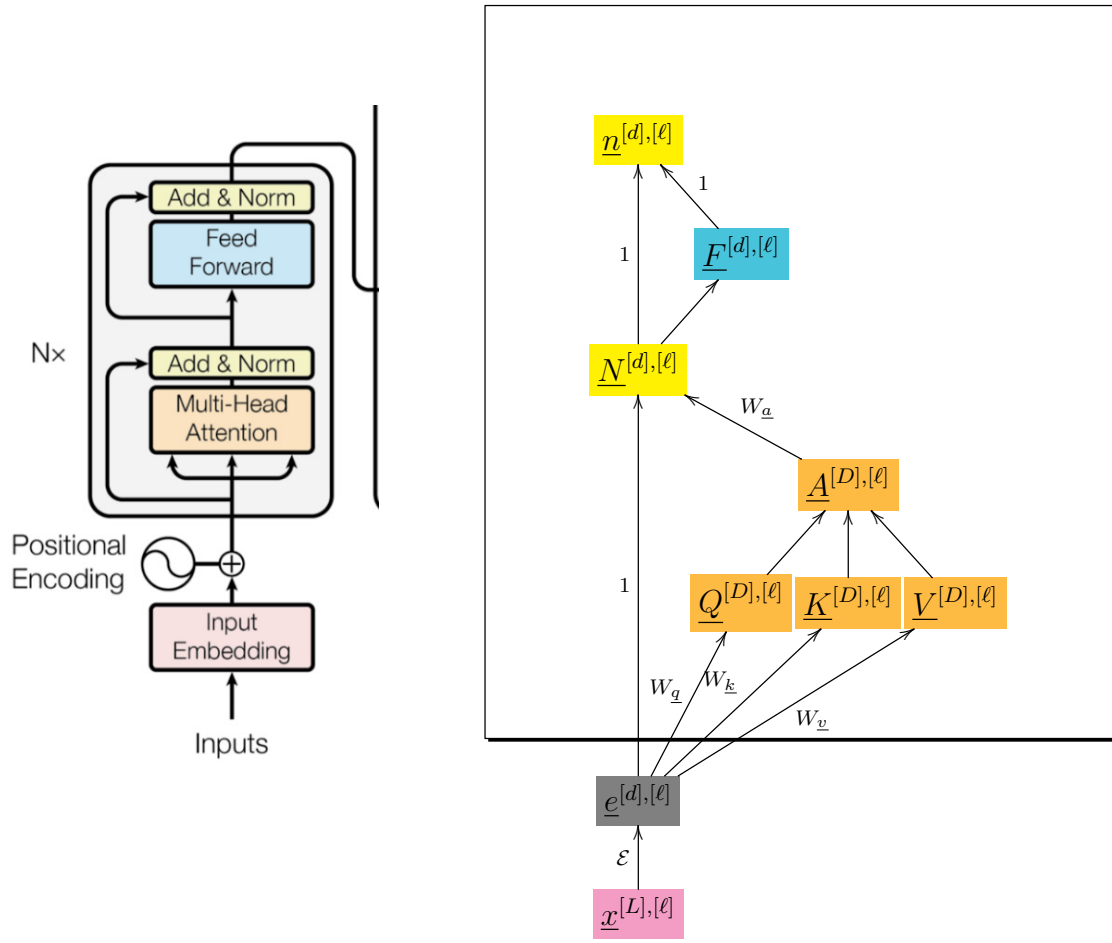


Figure 95.8: Encoder of Vanilla Transformer Net. N copies of the boxed part are connected in series.

$$N^{[d],[\ell]} = \text{normalize}(e^{[d],[\ell]} + W_{\underline{a}}^{[d],[D]} A^{[D],[\ell]}) \quad (95.48f)$$

$$Q^{[D],[\ell]} = W_{\underline{q}}^{[D],[d]} e^{[d],[\ell]} \quad (95.48g)$$

$$V^{[D],[\ell]} = W_{\underline{v}}^{[D],[d]} e^{[d],[\ell]} \quad (95.48h)$$

$$x^{[L],[\ell]} = \text{prior} \quad (95.48i)$$

95.3.4 Decoder

Fig.95.9 gives a bnet representation of the “Decoder” portion of Fig.95.4. The structural equations for that bnet, printed in blue, are as follows.

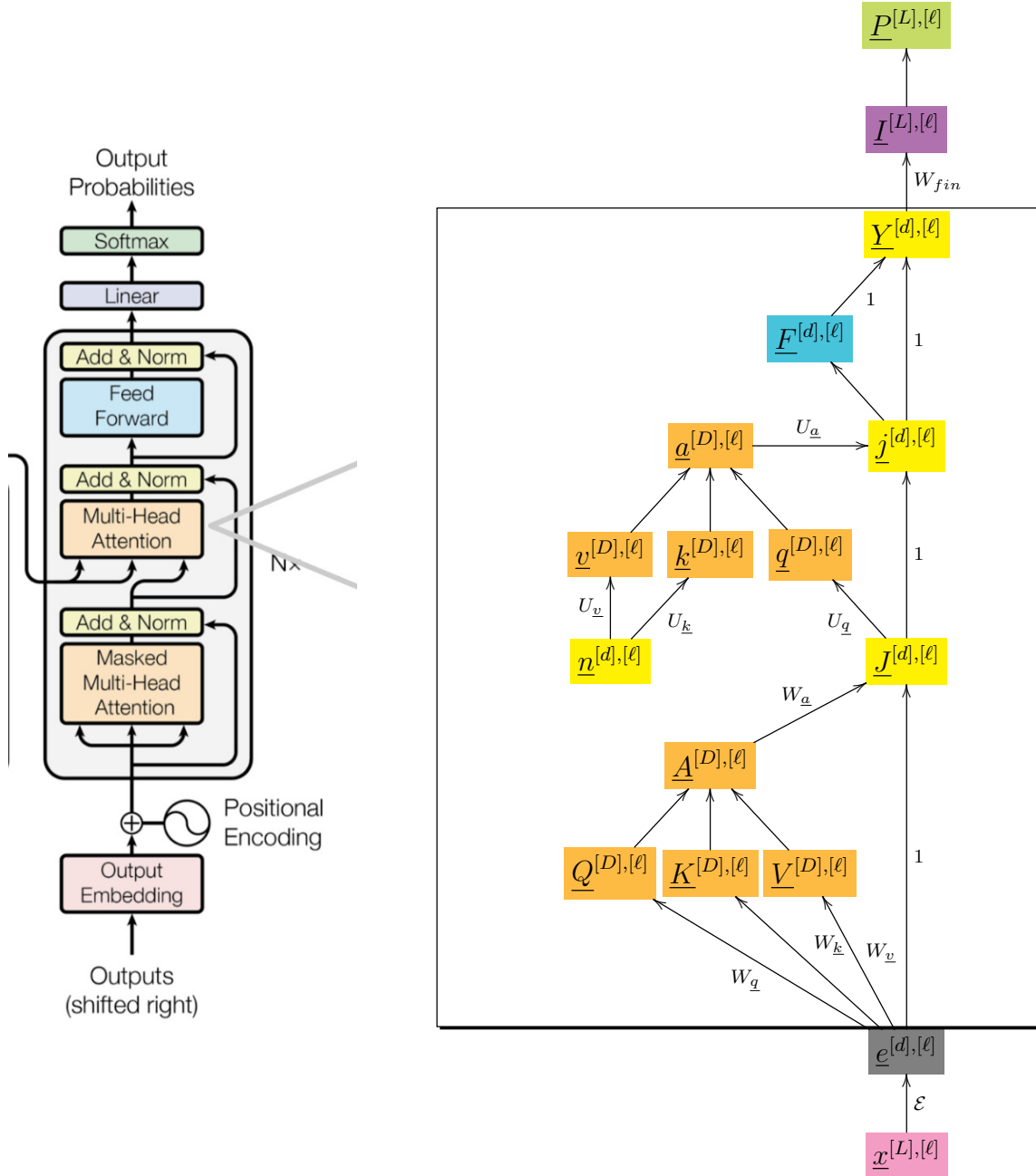


Figure 95.9: Decoder of Vanilla Transformer Net. N copies of the boxed part are connected in series.

$$a^{[D],[\ell]} = \text{Attention}(v^{[D],[\ell]}, k^{[D],[\ell]}, q^{[D],[\ell]}) \quad (95.49a)$$

$$A^{[D],[\ell]} = \text{Attention}(Q^{[D],[\ell]}, K^{[D],[\ell]}, V^{[D],[\ell]}) \quad (95.49b)$$

$$e^{[d],[\ell]} = \mathcal{E}^{[d],[L]} x^{[L],[\ell]} \quad (95.49c)$$

$$F^{[d],[\ell]} = \text{feed_forward_nn}(j^{[d],[\ell]}) \quad (95.49d)$$

$$I^{[L],[\ell]} = W_{fin}^{[L],[d]} Y^{[d],[\ell]} \quad (95.49e)$$

$$j^{[d],[\ell]} = \text{normalize}(U_{\underline{a}}^{[d],[D]} a^{[D],[\ell]} + J^{[d],[\ell]}) \quad (95.49f)$$

$$J^{[d],[\ell]} = \text{normalize}(W_{\underline{a}}^{[d],[D]} A^{[D],[\ell]} + e^{[d],[\ell]}) \quad (95.49g)$$

$$K^{[D],[\ell]} = W_{\underline{k}}^{[D],[d]} e^{[d],[\ell]} \quad (95.49h)$$

$$k^{[D],[\ell]} = U_{\underline{k}}^{[D],[d]} n^{[d],[\ell]} \quad (95.49i)$$

$$n^{[d],[\ell]} = \text{Prior coming from Encoder.} \quad (95.49j)$$

$$P^{[L],[\ell]} = \text{softmax}(I^{[L],[\ell]}) \quad (\sum_{\alpha \in [\ell]} P^{[L],\alpha} = 1) \quad (95.49k)$$

$$q^{[D],[\ell]} = U_{\underline{q}}^{[D],[d]} J^{[d],[\ell]} \quad (95.49l)$$

$$Q^{[D],[\ell]} = W_{\underline{q}}^{[D],[d]} e^{[d],[\ell]} \quad (95.49m)$$

$$V^{[D],[\ell]} = W_{\underline{v}}^{[D],[d]} e^{[d],[\ell]} \quad (95.49n)$$

$$v^{[D],[\ell]} = U_{\underline{v}}^{[D],[d]} n^{[d],[\ell]} \quad (95.49o)$$

$$x^{[L],[\ell]} = \text{prior, right shifted output} \quad (95.49p)$$

$$Y^{[d],[\ell]} = \text{normalize}(F^{[d],[\ell]} + J^{[d],[\ell]}) \quad (95.49q)$$

95.4 BERT

I used the the Wikipedia article on BERT, Ref[94] to write this section.

BERT (Bidirectional Encoder Representations from Transformer) is a realization of the Encoder half of the Vanilla tranet. One can either add a smaller NN to the output of BERT (this is called **fine-tuning**), or one can add a de-embedding layer to its output so that the total device takes word lists to word lists.

In the language of Bayesian Networks, fine-tuning is the same as using BERT as a prior probability. See Chapter 84 on sentence splitting for an example of BERT fine-tuning.

95.4.1 BERT parameter values

BERT comes in two sizes, base and large. See Table 95.1 for a listing of some BERT parameter values.

	BERT base	BERT large
ℓ , context window	512	512
L , vocab_size	30,522	30,522
d , hidden_size	768	1024
$n_{\underline{h}}$, num_attention_heads	12	16
Λ , num_hidden_layers	12	24
D' , intermediate_size	3,072	3,072
number of parameters	110M	340M

Table 95.1: Some hyper-parameter values for BERT base and BERT large

95.4.2 BERT Embedding

So far, we have described the embedding step as a single step from tokenization into words, to 1-hot vectors, to embedding vectors. There are other additional steps in the embedding process that we haven't described so far (namely, tokenization into subwords, adding special tokens, and padding). We would like to describe those additional steps now, in the context of the BERT model. Here is an example.

Let's consider a short sentence: "The cat is on the mat."

1. **Tokenization into words:** Tokenize the sentence into individual words:
"The", "cat", "is", "on", "the", "mat", "."
2. **Tokenization into subwords:** Further tokenize words into subword units using WordPiece tokenization or a similar method. For example:

The → *The*

cat → *ca, t*

is → *is*

on → *on*

the → *the*

mat → *mat*

. → *.*

Any subword not appearing in BERT's vocabulary is replaced by [UNK] for "unknown".

3. **Adding Special Tokens:** Add special tokens, such as [CLS] (classification) at the beginning and [SEP] (separator) at the end:
"[CLS]", "The", "ca", "t", "is", "on", "the", "mat", ".", "[SEP]"
4. **Padding:** If necessary, pad or truncate the sequence to a fixed length. Add padding tokens "[PAD]" to reach a specified sequence length.
5. **Embedding Matrix:** Create a tensor with 1-hot columns

$$x^{\beta, \alpha} = \delta(\beta, \beta(\alpha)) \quad (95.50)$$

where $\alpha \in [\ell]$, $\beta \in [L]$ and where $\beta(\alpha)$ is the location in the BERT vocab corresponding to token α in the padded string. Now multiply x times the previously discussed embedding matrix \mathcal{E} to get

$$e^{[d], [\ell]} = \mathcal{E}^{[d], [L]} x^{[L], [\ell]} \in \mathbb{R}^{d \times \ell} \quad (95.51)$$

This gives a vector in \mathbb{R}^d for each token α in the padded string. The matrix \mathcal{E} is pre-trained and captures contextual information and word similarities. It can also include positional embedding, as discussed before.

95.4.3 BERT training

BERT was trained⁶ simultaneously on two tasks.⁷

1. **language modeling:** 15% of tokens were selected for prediction. Those tokens selected for prediction were replaced by the [MASK] token 80% of the time, by a random word 10% of the time, and not replaced at all 10% of the time. The training objective was to predict the selected token given its context.
2. **next sentence prediction:** Given two spans of text, the model predicts if these two spans appeared sequentially in the training corpus, outputting either [IsNext] or [NotNext]. For example,
 - Given "[CLS] my dog is cute [SEP] he likes playing", the model should output token [IsNext].
 - Given "[CLS] my dog is cute [SEP] how do magnets work", the model should output token [NotNext]

⁶Sometimes this is called “pre-training” to distinguish it from the “training” of the smaller NN that is attached to the output of BERT when doing fine-tuning.

⁷This section on BERT training quotes Wikipedia Ref.[94] heavily.

Bibliography

- [1] Alan Agresti. *An introduction to categorical data analysis*. John Wiley & Sons, 2018.
- [2] Data Analytics and IIT Delhi Intelligence Research (DAIR) Group. Openie6. <https://github.com/dair-iitd/openie6>.
- [3] Elias Bareinboim, Jin Tian, and Judea Pearl. Recovering from selection bias in causal and statistical inference. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014. https://ftp.cs.ucla.edu/pub/stat_ser/r425.pdf.
- [4] Dan Bendel. Metropolis-Hastings: A comprehensive overview and proof. <https://similarweb.engineering/mcmc/>.
- [5] David Benkeser and Antoine Chambaz. A ride in targeted learning territory. <https://achambaz.github.io/tlride/tlride-book.pdf>.
- [6] Alexandra M Carvalho. Scoring functions for learning Bayesian networks. http://www.lx.it.pt/~asmc/pub/talks/09-TA/ta_pres.pdf.
- [7] Bo Chang. Copula: a very short introduction, article in Bo’s Blog. <https://bochang.me/blog/posts/copula/>.
- [8] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. <https://arxiv.org/abs/1603.02754>.
- [9] Victor Chernozhukov, Carlos Cinelli, Whitney Newey, Amit Sharma, and Vasilis Syrgkanis. Long story short: Omitted variable bias in causal machine learning. https://www.nber.org/system/files/working_papers/w30302/w30302.pdf.
- [10] Carlos Cinelli, Andrew Forney, and Judea Pearl. A crash course in good and bad controls. https://ftp.cs.ucla.edu/pub/stat_ser/r493.pdf.
- [11] Carlos Cinelli and Chad Hazlett. Making sense of sensitivity: Extending omitted variable bias. [https://carloscinelli.com/files/Cinelli%20and%20Hazlett%20\(2020\)%20-%20Making%20Sense%20of%20Sensitivity.pdf](https://carloscinelli.com/files/Cinelli%20and%20Hazlett%20(2020)%20-%20Making%20Sense%20of%20Sensitivity.pdf).

- [12] Scott Cunningham. *Causal inference: The mixtape*. Yale University Press, 2021. <https://mixtape.scunning.com/index.html>.
- [13] Robin J. Evans. Graphical methods for inequality constraints in marginalized DAGs. <https://arxiv.org/abs/1209.2978>.
- [14] Matheus Facure Alves. *Causal Inference for The Brave and True*. 2021. <https://matheusfacure.github.io/python-causality-handbook/landing-page.html>.
- [15] George Fei. Modeling uplift directly: Uplift decision tree with kl divergence and euclidean distance as splitting criteria. <https://www.aboutwayfair.com/tech-innovation/modeling-uplift-directly-uplift-decision-tree-with-kl-divergence-and-euclidean-distance-as-splitting-criteria>.
- [16] Charles Fox, Neil Girdhar, and Kevin Gurney. A causal Bayesian network view of reinforcement learning. <https://www.aaai.org/Papers/FLAIRS/2008/FLAIRS08-030.pdf>.
- [17] Bruno Gonçalves. Model testing and causal search. blog post <https://medium.com/data-for-science/causal-inference-part-vii-model-testing-and-causal-search-536b796f0384>.
- [18] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, David Warde-Farley Bing Xu, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. <https://arxiv.org/abs/1406.2661>.
- [19] Pierre Gutierrez and Jean-Yves G  rardy. Causal inference and uplift modelling: A review of the literature. In *Proceedings of The 3rd International Conference on Predictive Applications and APIs*, pages 1–13, 2017. <http://proceedings.mlr.press/v67/gutierrez17a.html>.
- [20] James Douglas Hamilton. *Time series analysis*. Princeton University Press, 2020.
- [21] Sebastian Haneuse and Andrea Rotnitzky. Estimation of the effect of interventions that modify the received treatment. *Statistics in medicine*, 32(30):5260–5277, 2013. Main:<https://sci-hub.se/10.1002/sim.5907>, Supplement: <https://onlinelibrary.wiley.com/action/downloadSupplement?doi=10.1002%2Fsim.5907&file=sim5907-sup-0001-Appendix.pdf>.
- [22] Christina Heinze-Deml. Causality, spring semester 2019 at ETH Zurich. https://stat.ethz.ch/lectures/ss19/causality.php#course_materials.
- [23] MA Hern  n and J Robins. *Causal inference: What if*. Boca Raton: Chapman & Hill/CRC, <https://www.hsph.harvard.edu/miguel-hernan/causal-inference-book/>.

- [24] Katherine Hoffman. An illustrated guide to modified treatment policies, part 1: Introduction and motivation, article in KHstats blog. <https://www.khstats.com/blog/lmtp/lmtp/>.
- [25] Katherine Hoffman. An illustrated guide to TMLE, article in KHstats blog. <https://www.khstats.com/blog/tmle/tutorial/>.
- [26] Cecil Huang and Adnan Darwiche. Inference in belief networks: A procedural guide. *International journal of approximate reasoning*, 15(3):225–263, 1996. <http://www.ar-tiste.com/Huang-Darwiche1996.pdf>.
- [27] Tommi S. Jaakkola and Michael I. Jordan. Variational probabilistic inference and the QMR-DT network. <http://arxiv.org/abs/1105.5462>.
- [28] Michael I. Jordan. course: Stat260-Bayesian modeling and inference, lecture date: February 8-2010, title: The conjugate prior for the Normal distribution. <https://people.eecs.berkeley.edu/~jordan/courses/260-spring10/lectures/lecture5.pdf>.
- [29] Chaitanya K. Joshi. Transformer (machine learning model). <https://graphdeeplearning.github.io/post/transformers-are-gnns/>.
- [30] Keshav Kolluru, Vaibhav Adlakha, Samarth Aggarwal, Mausam, and Soumen Chakrabarti. Openie6: Iterative grid labeling and coordination analysis for open information extraction. <https://arxiv.org/abs/2010.03147>.
- [31] Chung-Ming Kuan. Introduction to time series analysis, Fall 2014 lectures given at the Department of Finance, National Taiwan University. https://homepage.ntu.edu.tw/~ckuan/pdf/2014fall/Lec-TimeSeries_slide-Fall2014.pdf.
- [32] Steffen L Lauritzen and David J Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 50(2):157–194, 1988. <http://www.eecis.udel.edu/~shatkay/Course/papers/Lauritzen1988.pdf>.
- [33] Sergey Levine. Course CS 285 at UC Berkeley, Deep reinforcement learning. <http://rail.eecs.berkeley.edu/deeprlcourse/>.
- [34] Ang Li. Ph.D. Thesis, UCLA 2021. https://ftp.cs.ucla.edu/pub/stat_ser/r507.pdf.
- [35] Dimitris Margaritis. Learning Bayesian network model structure from data (thesis, 2003, Carnegie Mellon Univ). <https://apps.dtic.mil/sti/citations/ADA461103>.

- [36] Adam A Margolin, Ilya Nemenman, Katia Basso, Chris Wiggins, Gustavo Stolovitzky, Riccardo Dalla Favera, and Andrea Califano. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. In *BMC bioinformatics*, volume 7, page S7. Springer, 2006. <https://link.springer.com/article/10.1186/1471-2105-7-S1-S7>.
- [37] Samuele Mazzanti. Black-box models are actually more explainable than a logistic regression. <https://towardsdatascience.com/black-box-models-are-actually-more-explainable-than-a-logistic-regression-f263c22795d>.
- [38] Samuele Mazzanti. SHAP values explained exactly how you wished someone explained to you. <https://towardsdatascience.com/shap-explained-the-way-i-wish-someone-explained-it-to-me-ab81cc69ef30>.
- [39] Robert J. McEliece, David J. C. MacKay, and Jung-Fu Cheng. Turbo decoding as an instance of Pearl’s belief propagation algorithm. <http://authors.library.caltech.edu/6938/1/MCEieeejstc98.pdf>.
- [40] Scott Mueller, Ang Li, and Judea Pearl. Causes of effects: Learning individual responses from population data. *arXiv preprint arXiv:2104.13730*, 2021. <https://arxiv.org/abs/2104.13730>.
- [41] Brady Neal. Introduction to causal inference, Fall 2020, lectures and book. <https://www.bradyneal.com/causal-inference-course>.
- [42] Richard E Neapolitan. *Learning Bayesian networks*. Pearson Prentice Hall, 2004.
- [43] Andrew Ng. Lecture at deeplearning.ai on recurrent neural networks. <http://www.ar-tiste.com/ng-lec-rnn.pdf>.
- [44] Gregory Nuel. Tutorial on exact belief propagation in Bayesian networks: from messages to algorithms. <https://arxiv.org/abs/1201.4724>.
- [45] paperspace.com. PyTorch 101, Part 1: Understanding graphs, automatic differentiation and autograd. <https://blog.paperspace.com/pytorch-101-understanding-graphs-and-automatic-differentiation/>.
- [46] Judea Pearl. Linear models: A useful microscope for causal analysis. https://ftp.cs.ucla.edu/pub/stat_ser/r409-corrected-reprint.pdf.
- [47] Judea Pearl. Mediating instrumental variables. https://ftp.cs.ucla.edu/pub/stat_ser/r210.pdf.
- [48] Judea Pearl. On the testability of causal models with latent and instrumental variables. <https://arxiv.org/abs/1302.4976>.

- [49] Judea Pearl. Reverend Bayes on inference engines: A distributed hierarchical approach. <https://www.aaai.org/Papers/AAAI/1982/AAAI82-032.pdf>, 1982.
- [50] Judea Pearl. *Probabilistic Inference in Intelligent Systems*. Morgan Kaufmann, 1988.
- [51] Judea Pearl. The causal mediation formula—a guide to the assessment of pathways and mechanisms. *Prevention science*, 13(4):426–436, 2012. <https://apps.dtic.mil/sti/pdfs/ADA557663.pdf>.
- [52] Judea Pearl. *Causality: Models, Reasoning, and Inference, Second Edition*. Cambridge University Press, 2013.
- [53] Judea Pearl. Causal and counterfactual inference. *The Handbook of Rationality*, pages 1–41, 2019. https://ftp.cs.ucla.edu/pub/stat_ser/r485.pdf.
- [54] Judea Pearl and Elias Bareinboim. Transportability of causal and statistical relations: A formal approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, 2011. <https://ojs.aaai.org/index.php/AAAI/article/view/7861>.
- [55] Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. *Causal inference in statistics: A primer*. John Wiley & Sons, 2016.
- [56] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic Books, 2018.
- [57] Judea Pearl and James M Robins. Probabilistic evaluation of sequential plans from causal models with hidden variables. *arXiv preprint arXiv:1302.4977*, 2013. <https://arxiv.org/abs/1302.4977>.
- [58] Ashwin Rao and Tikhon Jelvis. *Foundations of Reinforcement Learning with Applications in Finance*. <https://stanford.edu/~ashlearn/RLForFinanceBook/book.pdf>.
- [59] ReliaSoft. System analysis reference. http://reliawiki.org/index.php/System_Analysis_Reference.
- [60] Piotr Rzepakowski and Szymon Jaroszewicz. Decision trees for uplift modeling with single and multiple treatments. *Knowledge and Information Systems*, 32(2):303–327, 2012. <https://link.springer.com/content/pdf/10.1007/s10115-011-0434-0.pdf>.
- [61] Scholarpedia. Granger causality. http://www.scholarpedia.org/article/Granger_causality.

- [62] Marco Scutari. bnlearn. <https://www.bnlearn.com/>.
- [63] Marco Scutari, Catharina Elisabeth Graafland, and José Manuel Gutiérrez. Who learns better Bayesian network structures: Accuracy and speed of structure learning algorithms. *International Journal of Approximate Reasoning*, 115:235–253, 2019. <https://arxiv.org/abs/1805.11908>.
- [64] Nitish Srivastava, G E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. <http://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf>.
- [65] StatQuest. XGBoost Parts 1-4 (videos). <https://statquest.org/video-index/>.
- [66] Russell Stuart and Norvig Peter. Artificial intelligence-a modern approach 3rd ed, 2016.
- [67] Masayoshi Takahashi. Statistical inference in missing data by MCMC and non-MCMC multiple imputation algorithms: Assessing the effects of between-imputation iterations. *Data Science Journal*, 16, 2017. <https://datascience.codata.org/articles/10.5334/dsj-2017-037/>.
- [68] theinvestorsbook.com. Pert analysis. <https://theinvestorsbook.com/pert-analysis.html>.
- [69] Jin Tian and Judea Pearl. Probabilities of causation: Bounds and identification. *Annals of Mathematics and Artificial Intelligence*, 28(1):287–313, 2000. https://ftp.cs.ucla.edu/pub/stat_ser/r271-A.pdf.
- [70] Robert R. Tucci. Bayesian networks (a.k.a. causal models, DAGs) and the passage of time. blog post in blog Quantum Bayesian Networks, <https://qbnets.wordpress.com/2021/07/16/bayesian-networks-aka-causal-models-dags-and-the-passage-of-time/>.
- [71] Robert R. Tucci. Bell’s inequalities for Bayesian statisticians. blog post in blog Quantum Bayesian Networks, <https://qbnets.wordpress.com/2008/09/19/bells-inequities-for-bayesian-statistician/>.
- [72] Robert R. Tucci. Goodness of causal fit. https://github.com/rrtucci/DAG_Lie_Detector.
- [73] Robert R. Tucci. JudeasRx. <https://github.com/rrtucci/JudeasRx>.
- [74] Robert R. Tucci. Mappa Mundi. https://github.com/rrtucci/mappa_mundi.
- [75] Robert R. Tucci. Quantum d-separation and quantum belief propagation. <https://arxiv.org/abs/2012.09635>.

- [76] Robert R. Tucci. Quantum Fog. <https://github.com/artiste-qb-net/quantum-fog>.
- [77] Robert R. Tucci. SCuMpy. <https://github.com/rrtucci/scumpy>.
- [78] Robert R. Tucci. SentenceAx. <https://github.com/rrtucci/SentenceAx>.
- [79] Robert R. Tucci. Shannon information theory without shedding tears over delta & epsilon proofs or typical sequences. <https://arxiv.org/abs/1208.2737>.
- [80] Robert R. Tucci. texnn. <https://github.com/rrtucci/texnn>.
- [81] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. <https://arxiv.org/abs/1706.03762>.
- [82] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl. Fault tree handbook nureg-0492. <https://www.nrc.gov/reading-rm/doc-collections/nuregs/staff/sr0492/>.
- [83] Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference (book). https://people.eecs.berkeley.edu/~wainwrig/Papers/WaiJor08_FTML.pdf.
- [84] Lilian Weng. The multi-armed bandit problem and its solutions. lilianweng.github.io/lil-log, <http://lilianweng.github.io/lil-log/2018/01/23/the-multi-armed-bandit-problem-and-its-solutions.html>, 2018.
- [85] Lilian Weng. What are diffusion models? lilianweng.github.io/lil-log, <https://lilianweng.github.io/posts/2021-07-11-diffusion-models/>, 2021.
- [86] Wikibooks. Control systems. https://en.wikibooks.org/wiki/Control_Systems.
- [87] Wikipedia. AdaBoost. <https://en.wikipedia.org/wiki/AdaBoost>.
- [88] Wikipedia. Autoregressive model. https://en.wikipedia.org/wiki/Autoregressive_model.
- [89] Wikipedia. Autoregressive moving-average model. https://en.wikipedia.org/wiki/Autoregressive%E2%80%93moving-average_model.
- [90] Wikipedia. Baum-Welsh algorithm. https://en.wikipedia.org/wiki/Baum%E2%80%93Welch_algorithm.
- [91] Wikipedia. Belief propagation. https://en.wikipedia.org/wiki/Belief_propagation.

- [92] Wikipedia. Berkson's paradox. https://en.wikipedia.org/wiki/Berkson%27s_paradox.
- [93] Wikipedia. Bernoulli distribution. https://en.wikipedia.org/wiki/Bernoulli_distribution.
- [94] Wikipedia. BERT language model. [https://en.wikipedia.org/wiki/BERT_\(language_model\)](https://en.wikipedia.org/wiki/BERT_(language_model)).
- [95] Wikipedia. Beta distribution. https://en.wikipedia.org/wiki/Beta_distribution.
- [96] Wikipedia. Beta function. https://en.wikipedia.org/wiki/Beta_function.
- [97] Wikipedia. Binary decision diagram. https://en.wikipedia.org/wiki/Binary_decision_diagram.
- [98] Wikipedia. Boolean algebra. https://en.wikipedia.org/wiki/Boolean_algebra.
- [99] Wikipedia. Bootstrap aggregating. https://en.wikipedia.org/wiki/Bootstrap_aggregating.
- [100] Wikipedia. Categorical distribution. https://en.wikipedia.org/wiki/Categorical_distribution.
- [101] Wikipedia. Chi-square distribution. https://en.wikipedia.org/wiki/Chi-square_distribution.
- [102] Wikipedia. Chow-Liu tree. https://en.wikipedia.org/wiki/Chow%E2%80%93Liu_tree.
- [103] Wikipedia. Cochran's theorem. https://en.wikipedia.org/wiki/Cochran%27s_theorem.
- [104] Wikipedia. Conjugate prior. https://en.wikipedia.org/wiki/Conjugate_prior.
- [105] Wikipedia. Copula. [https://en.wikipedia.org/wiki/Copula_\(probability_theory\)](https://en.wikipedia.org/wiki/Copula_(probability_theory)).
- [106] Wikipedia. Cramer-Rao bound. https://en.wikipedia.org/wiki/Cram%C3%A9r%E2%80%93Rao_bound.
- [107] Wikipedia. Cross-validation. [https://en.wikipedia.org/wiki/Cross-validation_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics)).

- [108] Wikipedia. Data processing inequality. https://en.wikipedia.org/wiki/Data_processing_inequality.
- [109] Wikipedia. Dirichlet distribution. https://en.wikipedia.org/wiki/Dirichlet_distribution.
- [110] Wikipedia. Errors in variables models. https://en.wikipedia.org/wiki/Errors-in-variables_models.
- [111] Wikipedia. Expectation maximization. https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm.
- [112] Wikipedia. F-distribution. <https://en.wikipedia.org/wiki/F-distribution>.
- [113] Wikipedia. Frisch-Waugh-Lovell theorem. https://en.wikipedia.org/wiki/Frisch%E2%80%93Waugh%E2%80%93Lovell_theorem.
- [114] Wikipedia. Functional derivative. https://en.wikipedia.org/wiki/Functional_derivative.
- [115] Wikipedia. Gamma distribution. https://en.wikipedia.org/wiki/Gamma_distribution.
- [116] Wikipedia. Gamma function. https://en.wikipedia.org/wiki/Gamma_function.
- [117] Wikipedia. Gated recurrent unit. https://en.wikipedia.org/wiki/Gated_recurrent_unit.
- [118] Wikipedia. Gibbs sampling. https://en.wikipedia.org/wiki/Gibbs_sampling.
- [119] Wikipedia. Granger causality. https://en.wikipedia.org/wiki/Granger_causality.
- [120] Wikipedia. Hidden Markov model. https://en.wikipedia.org/wiki/Hidden_Markov_model.
- [121] Wikipedia. Hoeffding's inequality. https://en.wikipedia.org/wiki/Hoeffding%27s_inequality.
- [122] Wikipedia. Importance sampling. https://en.wikipedia.org/wiki/Importance_sampling.
- [123] Wikipedia. Instrumental variables estimation. https://en.wikipedia.org/wiki/Instrumental_variables_estimation.

- [124] Wikipedia. Inverse transform sampling. https://en.wikipedia.org/wiki/Inverse_transform_sampling.
- [125] Wikipedia. Jackknife resampling. [https://en.wikipedia.org/wiki/Cross-validation_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics)).
- [126] Wikipedia. Junction tree algorithm. https://en.wikipedia.org/wiki/Junction_tree_algorithm.
- [127] Wikipedia. k-means clustering. https://en.wikipedia.org/wiki/K-means_clustering.
- [128] Wikipedia. Kalman filter. https://en.wikipedia.org/wiki/Kalman_filter.
- [129] Wikipedia. Kernel method. https://en.wikipedia.org/wiki/Kernel_method.
- [130] Wikipedia. Kernel perceptron. https://en.wikipedia.org/wiki/Kernel_perceptron.
- [131] Wikipedia. Laplace transform. https://en.wikipedia.org/wiki/Laplace_transform.
- [132] Wikipedia. Least squares. https://en.wikipedia.org/wiki/Least_squares.
- [133] Wikipedia. Legendre transformation. https://en.wikipedia.org/wiki/Legendre_transformation.
- [134] Wikipedia. Likelihood-ratio test. https://en.wikipedia.org/wiki/Likelihood-ratio_test.
- [135] Wikipedia. Linear regression. https://en.wikipedia.org/wiki/Linear_regression.
- [136] Wikipedia. Long short term memory. https://en.wikipedia.org/wiki/Long_short-term_memory.
- [137] Wikipedia. Markov blanket. https://en.wikipedia.org/wiki/Markov_blanket.
- [138] Wikipedia. Metropolis-Hastings method. https://en.wikipedia.org/wiki/Metropolis%E2%80%93Hastings_algorithm.
- [139] Wikipedia. Minimum spanning tree. https://en.wikipedia.org/wiki/Minimum_spanning_tree.
- [140] Wikipedia. Monte Carlo methods. https://en.wikipedia.org/wiki/Category:Monte_Carlo_methods.

- [141] Wikipedia. Moving-average model. https://en.wikipedia.org/wiki/Moving-average_model.
- [142] Wikipedia. Multinomial distribution. https://en.wikipedia.org/wiki/Multinomial_distribution.
- [143] Wikipedia. Multinomial theorem. https://en.wikipedia.org/wiki/Multinomial_theorem.
- [144] Wikipedia. Multivariate normal distribution. https://en.wikipedia.org/wiki/Multivariate_normal_distribution.
- [145] Wikipedia. Natural experiment. https://en.wikipedia.org/wiki/Natural_experiment.
- [146] Wikipedia. Non-negative matrix factorization. https://en.wikipedia.org/wiki/Non-negative_matrix_factorization.
- [147] Wikipedia. Ordinary least squares. https://en.wikipedia.org/wiki/Ordinary_least_squares.
- [148] Wikipedia. Program evaluation and review technique. https://en.wikipedia.org/wiki/Program_evaluation_and_review_technique.
- [149] Wikipedia. Random forest. https://en.wikipedia.org/wiki/Random_forest.
- [150] Wikipedia. Receiver operating characteristic. https://en.wikipedia.org/wiki/Receiver_operating_characteristic.
- [151] Wikipedia. Rejection sampling. https://en.wikipedia.org/wiki/Rejection_sampling.
- [152] Wikipedia. Score test. https://en.wikipedia.org/wiki/Score_test.
- [153] Wikipedia. Signal flow graph. https://en.wikipedia.org/wiki/Signal-flow_graph.
- [154] Wikipedia. Simple linear regression. https://en.wikipedia.org/wiki/Simple_linear_regression.
- [155] Wikipedia. Simpson's paradox. https://en.wikipedia.org/wiki/Simpson's_paradox.
- [156] Wikipedia. Spring system. https://en.wikipedia.org/wiki/Spring_system.

- [157] Wikipedia. Student's t-distribution. https://en.wikipedia.org/wiki/Student%27s_t-distribution.
- [158] Wikipedia. Support vector machine. https://en.wikipedia.org/wiki/Support-vector_machine.
- [159] Wikipedia. Survival analysis. https://en.wikipedia.org/wiki/Survival_analysis.
- [160] Wikipedia. Time series. https://en.wikipedia.org/wiki/Time_series.
- [161] Wikipedia. Transfer learning. https://en.wikipedia.org/wiki/Transfer_learning.
- [162] Wikipedia. Transformer (machine learning model). [https://en.wikipedia.org/wiki/Transformer_\(machine_learning_model\)](https://en.wikipedia.org/wiki/Transformer_(machine_learning_model)).
- [163] Wikipedia. Uplift modelling. https://en.wikipedia.org/wiki/Uplift_modelling.
- [164] Wikipedia. Variational Bayesian methods. https://en.wikipedia.org/wiki/Variational_Bayesian_methods.
- [165] Wikipedia. Vector autoregression. https://en.wikipedia.org/wiki/Vector_autoregression.
- [166] Wikipedia. Viterbi algorithm. https://en.wikipedia.org/wiki/Viterbi_algorithm.
- [167] Wikipedia. Wald test. https://en.wikipedia.org/wiki/Wald_test.
- [168] Wikipedia. Z-transform. <https://en.wikipedia.org/wiki/Z-transform>.
- [169] Hao Wu and Zhaohui Steve Qin. course notes, BIOS731: Advanced statistical computing, 2016 Emory Univ. <http://web1.sph.emory.edu/users/hwu30/teaching/statcomp/statcomp.html>.
- [170] Ronghui (Lily) Xu. Lecture notes, MATH 284, Spring 2020, Survival analysis. <https://mathweb.ucsd.edu/~rxu/math284/>.
- [171] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Understanding belief propagation and its generalizations, Mitsubishi Technical Report tr-2001-22. <https://merl.com/publications/docs/TR2001-22.pdf>.