MIPaaL: Mixed Integer Program as a Layer - Supplemental Information

Aaron Ferber,[1] Bryan Wilder,[2] Bistra Dilkina,[1] Milind Tambe[2]
[1]University of Southern California, [2]Harvard John A. Paulson School of Engineering and Applied Sciences
aferber@usc.edu, bwilder@g.harvard.edu, dilkina@usc.edu, milind_tambe@harvard.edu

Implementation Details

Experiments are run on a cluster of five identical 32-core machines with Intel 2.1 GHz processors and 264 GB of memory. We use the C API of IBM ILOG CPLEX 12.6.1 to generate and record cutting planes during training. Neural networks are trained and tested with the PyTorch Python API (Paszke et al. 2017) and evaluated with the CPLEX's Python API. We average over 5 training and testing iterations per problem setting with different seeds to evaluate the given approaches. This results in 180 portfolio optimization, 55 diverse bipartite matching, and 95 weighted knapsack instances to compute testing metrics. The neural network architectures are fixed for a given problem distribution, and are selected from grid search using MIP decision quality on the validation set of the *TwoStage model*. The models are trained with Adam (Kingma and Ba 2014) with learning rate 0.01 and l2 normalization coefficient of 0.01. Details on model architectures and evaluation setup can be found in the appendix.

Predictive architectures

The neural networks for the portfolio optimization problems consist of two fully-connected layers with 100 nodes each, followed by batch normalization (Ioffe and Szegedy 2015), LeakyRelu as a nonlinearity (Maas, Hannun, and Ng 2013), and dropout with probability 0.5 as proposed and suggested in (Srivastava et al. 2014). As portfolio optimization and knapsack are built on regression problems, we add a linear layer on the output of the predictive component. The architecture for matching is similar although it has only one layer and uses a sigmoid activation function on the output of the predictive component since we perform edge prediction. These architectures were selected from grid search using MIP decision quality on the validation set of just the standard two stage model. The grid search was done considering models with between one and three layers, ReLU, LeakyReLU, and Sigmoid activation functions, and either 100 or 200 nodes per layer. We additionally, ran experiments disregarding dropout and batch normalization and found that without either the networks were unable to generalize. The models are trained with the Adam optimizer (Kingma and Ba 2014) with learning rate 0.01 and l2 normalization coefficient of 0.01.

Decision Quality: Pairwise Win-Loss Comparison

To further tease out the benefit of using variants of MIPaaL on a case-by-case basis, we compare decision quality in terms of win/loss rates in Table 1. This table shows the percent of instances the given approach on the left won/lost to the approach on the top, matching instances based on optimization problem and seed. Additionally, we compare the means of these values using a one-sided t-test with significance level 0.05. Asterisks indicate statistically significant results.

As is apparent in Table 1, the MIPaaL method tends to outperform the other methods on all problem settings on an individual instances basis. Also useful to note is that the RootLP method beats the MIPaaL-100 approach in both the portfolio optimization settings, potentially indicating that limiting cut generation too much can be detrimental to the performance of the overall system. Additionally, in the diverse matching setting, the standard two stage approach does comparably to the k-cut decision-focused methods which stop cut generation early. This suggests that while the diverse matching problem is similar to the original bipartite matching LP, adding a few diversity constraints breaking the integrality property of the LP relaxation can remove gains made from decision-focused learning unless the full MIPaaL layer is used.

Extended Timing Results

We present full timing results in Table 2. Times for MIP layers are relatively similar within the same problem type. Across problem types, problem instances on DAX and on knapsack solve quickly and take little time for KKT computation. However, on SP500 and matching, much more time is dedicated to computing the backward pass through the MIP layer for all approaches. In our experiments, the decision-focused approaches have timings correlated with the number of cuts generated, with RootLP generating 0 cuts. Lastly, the TwoStage approach is clearly fastest as the approach benefits not only from polynomial time inference but also from vectorization and parallelization across instances in a batch. Overall the optimization layer approaches do suffer in training time but make up for it in decision quality. Looking at the distributions, MIPaaL-100 performs similarly to RootLP in terms of equality percentage and difference to the optimal

Table 1: Decision quality per instance win / loss percentages, with * indicating decision performance of winner is significantly better according to paired t test with 0.05 significance level. MIPaaL reliably improves or ties in these settings.

		MIPaaL-1000	MIPaaL-100	RootLP	TwoStage
SP500	MIPaaL-Full	57.2 / 42.3 *	87.2 / 12.8 *	82.8 / 17.2 *	90.6 / 9.4 *
	MIPaaL-1000		84.4 / 15.6 *	80.0 / 20.0 *	85.0 / 15.0 *
	MIPaaL-100			27.8 / 72.2 *	52.8 / 47.2
	RootLP				69.4 / 30.6 *
DAX	MIPaaL-Full	66.7 / 33.3 *	86.6 / 13.4 *	88.9 / 11.1 *	84.4 / 15.6 *
	MIPaaL-1000		81.1 / 18.9 *	85.6 / 14.4 *	73.9 / 26.1 *
	MIPaaL-100			69.4 / 30.6 *	43.3 / 56.7
	RootLP				28.3 / 71.7 *
Diverse Matching	MIPaaL-Full	11.7 / 3.9 *	15.6 / 2.8 *	15.0 / 3.9 *	12.2 / 5.6 *
	MIPaaL-1000		11.7 / 3.9	9.4 / 8.3	10.0 / 7.2
	MIPaaL-100			3.9 / 11.1	5.0 / 8.9
	RootLP				7.2 / 8.3

Table 2: Timing results comparison. We give the average time to complete one epoch as well as the average time spent in forward and backward components of MIPaaL layers for a single MIP instance. Percentages of the total epoch time are also given for forward and backward passes through the MIP layer. Times are all given in seconds.

		Epoch (s)	MIP Forward	MIP Backward	MIP Forward %	MIP Backward %
SP500	MIPaaL	485.90	0.2615	1.7180	3.88%	25.46%
	MIPaaL-1000	483.02	0.2606	1.6091	3.88%	23.99%
	MIPaaL-100	482.57	0.2576	1.5932	3.84%	23.77%
	RootLP	480.57	0.2524	1.5749	3.78%	23.59%
	TwoStage	34.51	N/a	N/a	N/A	N/A
DAX	MIPaaL	47.06	0.1021	0.0218	15.63%	3.34%
	MIPaaL-1000	45.38	0.1014	0.0164	16.09%	2.59%
	MIPaaL-100	44.86	0.1009	0.0162	16.20%	2.60%
	RootLP	44.71	0.1002	0.0153	16.14%	2.47%
	TwoStage	19.94	N/a	N/a	N/A	N/A
Matching	MIPaaL	603.76	0.8040	12.4155	2.13%	32.90%
	MIPaaL-1000	595.45	0.7322	12.1961	1.97%	32.77%
	MIPaaL-100	585.79	0.7107	11.8781	1.94%	32.44%
	RootLP	584.76	0.7032	11.8204	1.92%	32.34%
	TwoStage	13.91	N/a	N/a	N/A	N/A
Knapsack	MIPaaL	207.86	0.1599	0.0135	4.31%	0.36%
	MIPaaL-1000	206.48	0.1579	0.0133	4.28%	0.36%
	MIPaaL-100	203.89	0.1570	0.0130	4.31%	0.36%
	RootLP	159.70	0.1198	0.0113	4.20%	0.40%
	TwoStage	0.99	N/a	N/a	N/A	N/A

integral solution in all settings except knapsack and DAX, two settings for which RootLP gives solutions comparatively far away from the optimal integral solution.

Integer Infeasibility Metrics

we give infeasibility metrics of our approaches in Table 3. Across all settings, MIPaaL gives solutions closer to the optimal integral solution demonstrating that the resulting LP closely characterizes the region around the optimal solution. Additionally, we use the average difference

Bipartite matching

Problem

We run experiments on a version of the bipartite matching problem used in (Wilder, Dilkina, and Tambe 2019) which requires additional diversity constraints on the proposed matching. The matching problem is done on graphs sampled from the CORA citation network dataset (Sen et al. 2008) which consist of nodes and edges representing publications and citations respectively. In addition, we use information present about a given paper's field of study.

Data specification

The full network we consider consists of 2708 nodes which are partitioned into 27 matching instances using metis (Karypis and Kumar 1998). Each instance is a bipartite matching problem on a complete bipartite graph with 50 nodes on each side of the graph. The 100 nodes in each instance are divided to maximize the number of edges between the two partitions. To ensure the diversity of our recommendations, we require a minimum $p = 25\%$ percentage of the suggested pairings belong to distinct fields of study and similarly impose a constraint that a minimum percentage $q = 25\%$ of suggested citations belong to the same field of study. The predicted values in this case are the edge weights which correspond to whether one paper actually cites another. The node representations in this case are binary feature vectors which correspond to whether a given word from a lexicon of 1433 words appears in the paper. Edge values are predicted based on the concatenation of the node features on the edge endpoints.

Optimization model

Given: two sets of nodes representing publications N_1, N_2, weights on pairs of nodes $c_{i,j} \forall i \in N_1, j \in N_2$ corresponding to how likely it is that one paper will cite another, same field indicator $m_{i,j} = \{1 \text{ if i and j are in the same field}, 0 \text{ otherwise}\}$
Goal: find matching of nodes such that 1) each node is matched at most once, 2) at least $p \in [0, 1]$ proportion of selected edges connect nodes of the same field ($m_{i,j} = 1$) 3) at least $q \in [0, 1]$ proportion of selected edges connect nodes of different fields ($m_{i,j} = 0$)
Decision Variables: $x_{i,j} \in 0, 1 \forall (i, j) \in N_1 \times N_2$ corresponding to whether we use edge (i, j) in the matching or not.

Formulation:

$$\begin{aligned}
\max_x \textstyle\sum_{i,j} c_{i,j} x_{i,j} & & \\
\text{s.t.} \textstyle\sum_j x_{i,j} &\leq 1 & \forall i \in N_1 \\
\textstyle\sum_i x_{i,j} &\leq 1 & \forall j \in N_2 \\
\textstyle\sum_{i,j} m_{i,j} x_{i,j} &\geq p \textstyle\sum_{i,j} x_{i,j} & \\
\textstyle\sum_{i,j} (1 - m_{i,j}) x_{i,j} &\geq q \textstyle\sum_{i,j} x_{i,j} & \\
x_{i,j} &\in 0, 1 & \forall i \in N_1, j \in N_2
\end{aligned} \tag{1}$$

Overall this problem has $|N_1| \times |N_2|$ decision variables, and $|N_1| + |N_2| + 2$ constraints. In our setting, we had $p = q = 0.25$ to validate our experiments as this setting resulted in the problem not simply being solved too easily while yielding feasible regions.

Portfolio Optimization Specifics

Problem

In the combinatorial portfolio optimization problem we are given features of assets, as well as estimates of rates of returns and our objective is to invest money among these assets to maximize the expected return of the portfolio while ensuring the portfolio meets some requirements. In particular, our model is similar to that investigated in (Bertsimas, Darnell, and Soucy 1999) in that we don't want our proposed portfolio to be too different from a "tracked" portfolio, as well as limiting the number of items in a given sector so that we do not invest too much of our money in one place. Similarly, we don't want to exercise too many trades so we limit the number of trades we will need to make to get to the proposed portfolio. Finally, we need to be able to execute our trades so we limit assets based on how often they are traded.

Data specification

We use historical price and volume data downloaded from the Quandl WIKI dataset (Quandl 2019). We train our model on monthly data collected from January 2005 to December 2010, validate on data collected from January 2011 to November 2013 and test our model based on data from December 2013 to November 2016 resulting in 72 time periods used for training, 35 for validation and 36 for testing. We split the data temporally to ensure that data from the future isn't used to inform predictions in the current time period. The 11 features used are made up of historical price averages and rates of return over different time horizons, which are commonly used indicators for simple trading strategies. Furthermore, to evaluate the generality of the approach we evaluate on two market indices: the SP500 and DAX. The SP500 is an index of 505 large companies in the United States which are widely representative of the US trading market. The DAX is composed of 30 large German companies and is much smaller than the SP500, being somewhat less representative of its respective market as a whole, but still representative of the central publicly traded companies in terms of trading volume and amount invested in those companies.

We use the following 11 indicators commonly used in simple momentum-based and mean-reversion trading strategies (Conrad and Kaul 1998):

Table 3: Infeasibility metrics measure how often an optimal integer solution is obtained. We measure infeasibility and difference to the optimal integral solution with two metrics: the percent of binary variables equal to their values in the optimal solution, and the average absolute difference between the solution to the final obtained cutting plane relaxation and the optimal integral solution obtained through branch and bound.

	SP500		DAX		Matching		Knapsack	
	% Eq	Avg Diff	% Eq	Avg Diff	% Eq	Avg Diff	% Eq	Avg Diff
MIPaaL	89.20%	0.10	70.40%	0.13	88.35%	0.12	97.13%	0.10
MIPaaL-1000	45.91%	0.28	69.28%	0.18	87.81%	0.14	93.84%	0.19
MIPaaL-100	28.36%	0.29	67.11%	0.41	87.64%	0.22	91.97%	0.24
RootLP	23.59%	0.29	36.63%	0.45	86.43%	0.24	69.68%	0.34

percentage increase of price from previous 5 time periods
percentage increase of price from previous year
percentage increase of price from previous quarter
mean of price percentage increase over previous year
variance of price percentage increase over previous year
mean of price percentage increase over previous quarter
variance of price percentage increase over previous quarter

We randomly sample 60 indices from the SP500 for SP-30^a and SP-30^b, these indices are:
SP-30^a = LRCX, PHM, WY, SPG, EMN, CME, AEP, F, CAG, FISV, WBA, XOM , NVDA, ETN, MDT, FL, HBAN, FFIV, BLK, IPG, EXPD, IRM, PH , DLTR, COST, NBL, INCY, CSX]
SP-30^b = [DVA, DE, BAC, KLAC, ADBE, FIS, IT, KR, FMC, HOG, SHW, RE , ETR, BK, ACN, NWL, ESS, VMC, C, EW, IR, SWKS, SNPS, ARE , SCHW, WEC, IVZ, SLB

Indices for the SP-50, SP-100, and SP-200 are:
SP-50 = VLO, EIX, RSG, PKG, CMI, AMZN, CHRW, ATVI, ADS, RHI, DVN , WMB, ILMN, LOW, SIVB, T, NTRS, HCP, ALXN, CPB, MRO, MU, CB , HES, APA, VFC, CHD, CMCSA, ALK, PBCT, BBY, MNST, UDR, CTXS , AZO, XRX, SBAC, M, MTD, COP, UTX, KO, MCO, TGT, CELG, HSIC , WYNN, YUM, ECL, ABT

SP-100 = RTN, K, ROST, WAB, MMM, WMT, JPM, MMC, AMD, BA, NEM, JBHT , STI, ITW, NKTR, ETFC, PNW, HSY, EL, MLM, UPS, FRT, ES, ROL , CMS, MAC, PVH, TSN, ANSS, SJM, DISH, LUV, MOS, TTWO, SEE , PKI, BAX, AMG, EOG, AMGN, CCI, EA, AGN, HP, HAL, AXP, SRE , INTU, AOS, APH, ROK, ABMD, CINF, UNM, ZION, HRL, ADSK, MSFT , JNJ, EBAY, ADI, EXC, CVS, PEG, CVX, PNR, PAYX, CPRT, DHR , JCI, SYK, HST, MO, INTC, GD, MCHP, PFG, PG, QCOM, STX, HAS , WM, CL, CDNS, BF_B, HIG, REGN, COG, RL, GPS, APD, APC, GPC , TSCO, JKHY, FAST, NOV, TMK, AVY, PGR

SP-200 = SYMC, WDC, CSCO, STZ, NUE, MAA, ARNC, DGX, EQIX, PWR, ED , LLY, AIV, ISRG, FITB, EQR, BXP, ORLY, MHK, JEC, UNP, TXT , IDXX, WHR, OMC, KEY, NI, FLIR, SNA, MCK, PXD, MCD, XEC, LEG , KSU, TSS, IP, AVB, RF, GPN, AFL, UNH, DTE, NEE, ZBH, NDAQ , DRE, MSI, VRSN, A, XEL, MAS, LKQ, FDX, LMT, WFC, NRG, RHT , COF, RMD, SLG, TROW, ANTM, BRK_B, PSA, RJF, BSX, AJG, FLR , BLL, AES, ORCL, MAR, HD, GILD, RCL, TIF, PRU, SWK, TMO, NOC , NKE, IFF, DHI, STT, AMT, TJX, PRGO, GS, NFLX, GLW, PFE, DOV , OKE, PLD, VZ, LNC, PNC, CTSH, JNPR, LEN, ABC, CI, CCL, LH , FCX, FLS, PCAR, HRS, EMR, GRMN, AKAM, BBT, PEP, OXY, KSS, MS , URI, AEE, NSC, CAH, VTR, O, USB, SYY, FE, TXN, BDX, BWA , CMA, COO, NTAP, SBUX, GWW, CNP, HOLX, CNC, D, TAP, MET, AAP , TFX, IBM, XLNX, LLL, AMAT, HUM, AON, DIS, KMB, BEN, GIS, ADM , MXIM, PPL, HPQ, ALL, VRTX, L, KIM, DOW, HON, DRI, AAPL, XRAY , MKC, MRK, HFC, FOX, CTAS, REG, HRB, KMX, SO, MYL, CLX, CERN , ALB, BIIB, LNT, BMY, MAT, DUK, MTB, AME, LB, GE, VNO, ROP , EFX, AIG, JWN, CAT, UHS, WAT, ADP, ATO, VAR, FOXA, MDLZ

Optimization model

We specify the optimization model used for the portfolio optimization task.

For our applications, we set the ticket budget $\mathrm{B}_{\text{ticket}}$, and B_{name} to be half of the number securities considered, 200 for SP500, and 15 for DAX and SP-30. We set the sector budget $\mathrm{B}_{\text{sector}}$ to be 0.1, in that for any given sector we can change at most 10% of the portfolio weight into or out of that sector. Overall we found that this gave us a balance of reasonable problems that weren't trivial to solve but still had a non-empty feasible region.

Given: Set of n assets $i = 1, \ldots, n$, set of sectors S represented as a partitioning of the n assets
Ticket (trading) limit $\mathrm{B}_{\text{ticket}}$
Name (unique item changes) limit B_{name}
Sector deviation limit $\mathrm{B}_{\text{sector}}$
For each asset $i = 1, \ldots, n$:
Expected returns α_i
Trading volume vol(i)
Indicator whether asset i belongs to sector s
$M_s(i) \equiv i \in s$
Initial portfolio asset weight $w_0(i)$
Target portfolio asset weight $w_t(i)$

Goal: find final portfolio weights $w_f(i)$ which is close to the target portfolio w_t(i) in terms of weight, doesn't incur too much cost from $w_0(i)$ to execute, respects budget constraints in terms of sector exposure, cardinality, and trading limits, and ensuring all weights add up to 1

Decision variables: final weight in asset i $w_f(i) \in [0,1] \forall i = 1, \ldots, n$
auxiliary ticket-counting variables $z_1(i), z_2(i), z_3(i)$
auxiliary indicator of whether asset i is used $y_{\text{names}}(i)$
auxiliary indicator of whether asset i changes weight in the portfolio y_{tickets}
auxiliary variable $f(i)$ representing absolute value of weight change from original, keeping track of tickets bought
auxiliary variable $y(i)$ representing absolute value of deviation from target
auxiliary variable $x(s)$ representing absolute value of sector weight change
auxiliary variables $z_1(i), z_2(i), z_3(i)$ to put weights in three different compartments of a piecewise linear function related to volume.

Formulation:

$$\begin{array}{rll}
\max \sum_{i=1}^{n} \alpha_i w(i) & & \\
\text{s.t.} \sum_{i=1}^{n} w_f(i) = 1 & & \\
\sum_{s \in S} x(s) \leq \text{B}_{\text{sector}} & & \\
\sum_{i=1}^{n} \text{y}_{\text{names}}(i) \leq \text{B}_{\text{name}} & & \\
\sum_{i=1}^{n} y_{\text{tickets}}(i) \leq \text{B}_{\text{ticket}} & & \\
w_f(i) - w_t(i) \leq y(i) & & \forall \text{i} = 1..\text{n} \\
-(w_f(i) - w_t(i)) \leq y(i) & & \forall \text{i} = 1..\text{n} \\
w_f(i) - w_0(i) \leq f(i) & & \forall \text{i} = 1..\text{n} \\
-(w_f(i) - w_0(i)) \leq f(i) & & \forall \text{i} = 1..\text{n} \\
\sum_{i=1}^{n} M_s(i)(w_f(i) - w_t(i)) \leq x(s) & & \forall s \in S \\
-\sum_{i=1}^{n} M_s(i)(w_f(i) - w_t(i)) \leq x(s) & & \forall s \in S \\
w_f(i) \leq \text{y}_{\text{names}}(i) & & \forall \text{i} = 1..\text{n} \\
f(i) \leq y_{\text{tickets}}(i) & & \forall \text{i} = 1..\text{n} \\
f(i) = \sum_{u=1}^{3} z_u(i) & & \forall \text{i} = 1..\text{n} \\
0 \leq z_1(i) \leq 0.1\,\text{vol(i)} & & \forall \text{i} = 1..\text{n} \\
0 \leq z_2(i) \leq 0.2\,\text{vol(i)} & & \forall \text{i} = 1..\text{n} \\
0 \leq z_3(i) \leq 0.2\,\text{vol(i)} & & \forall \text{i} = 1..\text{n} \\
w_f(i), f(i), y(i) \geq 0 & & \forall \text{i} = 1..\text{n} \\
z_1(i), z_2(i), z_3(i) \geq 0 & & \forall \text{i} = 1..\text{n} \\
\text{y}_{\text{names}}(i), y_{\text{tickets}}(i) \in \{0,1\} & & \forall \text{i} = 1..\text{n} \\
x(s) \geq 0 & & \forall s \in S \\
x_{i,j} \in 0,1 & & \begin{array}{l}\forall i \in N_1 \\ j \in N_2\end{array}
\end{array} \tag{2}$$

Overall this problem has $|S| + 6n$ continuous decision variables, $|2n|$ binary decision variables, and $10n + |2S| + 4$ constraints.

Energy-based Knapsack

Knapsack instances consider that we are given a weight budget B and n items, each with value v_i and weight w_i. The objective is to select items maximizing the value but ensuring the budget limit on total weight is satisfied. This can be written as the simple MIP,

$$\begin{array}{l}
\max \sum_{i=1}^{n} v_i x_i \\
\text{s.t.} \sum_{i=1}^{n} w_i x_i \leq B \\
x_i \in \{0,1\} \forall\, i = 1, \ldots, n
\end{array} \tag{3}$$

Data specification

Our knapsack benchmark corresponds to a unit commitment problem where we want to sell energy at half-hour intervals over the day with different revenue at different times of the day and an operating budget with known varying difficulty of operating at different intervals. We want to maximize the revenue we get from the day schedule but we need to predict what the half-hour prices will be. We use real-world energy-aware scheduling data from (Demirovic et al. 2019) (obtained from the authors), which consist of historical energy prices at different half-hour intervals over the day along with 8 features of the given intervals (including price projections stated ahead of time) from the ICON energy-aware scheduling competition. Each MIP represents an energy-based problem over 48 time steps. The weights per half-hour interval are drawn uniformly between 0 and 1, and the budget is 10% of the total weight.

References

Bertsimas, D.; Darnell, C.; and Soucy, R. 1999. Portfolio construction through mixed-integer programming at grantham, mayo, van otterloo and company. *Interfaces* 29(1):49–66.

Conrad, J., and Kaul, G. 1998. An anatomy of trading strategies. *The Review of Financial Studies* 11(3):489–519.

Demirovic, E.; Stuckey, P. J.; Bailey, J.; Chan, J.; Leckie, C.; Ramamohanarao, K.; and Guns, T. 2019. Predict+optimise with ranking objectives: Exhaustively learning linear functions. In *IJCAI*.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 448–456.

Karypis, G., and Kumar, V. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing* 20(1):359–392.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Maas, A. L.; Hannun, A. Y.; and Ng, A. Y. 2013. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on International Conference on Machine Learning (ICML)*.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; and Yang, E. e. a. 2017. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*.

Quandl. 2019. Various end-of-day data. https://www.quandl.com/.

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI magazine* 29(3):93–93.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR* 15(1):1929–1958.

Wilder, B.; Dilkina, B.; and Tambe, M. 2019. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *AAAI*.

.