Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Data-Driven Approximations to NP-Hard Problems

Anton Milan, S. Hamid Rezatofighi, Ravi Garg, Anthony Dick, Ian Reid

School of Computer Science, The University of Adelaide, Australia
{firstname.lastname}@adelaide.edu.au

Abstract

There exist a number of problem classes for which obtaining the exact solution becomes exponentially expensive with increasing problem size. The quadratic assignment problem (QAP) or the travelling salesman problem (TSP) are just two examples of such NP-hard problems. In practice, approximate algorithms are employed to obtain a suboptimal solution, where one must face a trade-off between computational complexity and solution quality. In this paper, we propose to learn to solve these problem from *approximate* examples, using recurrent neural networks (RNNs). Surprisingly, such architectures are capable of producing highly accurate solutions at minimal computational cost. Moreover, we introduce a simple, yet effective technique for improving the initial (weak) training set by incorporating the objective cost into the training procedure. We demonstrate the functionality of our approach on three exemplar applications: marginal distributions of a joint matching space, feature point matching and the travelling salesman problem. We show encouraging results on synthetic and real data in all three cases.

Introduction

Artificial neural networks have emerged in a wide variety of applications, successfully tackling problems previously considered too complex for existing techniques. Examples of such recent breakthroughs include speech recognition (Graves, Mohamed, and Hinton 2013), image classification (Krizhevsky, Sutskever, and Hinton 2012), and superhuman performance in video and board games (Mnih et al. 2015; Silver et al. 2016). Recently, deep neural networks have also been applied to seemingly complex tasks like predicting the output of short computer programs (Zaremba and Sutskever 2014) or computing the shortest path of the travelling salesman problem (TSP) (Vinyals, Fortunato, and Jaitly 2015). Inspired by this recent work, we make two important contributions.

Our first contribution is an LSTM-based architecture that is capable of learning to predict complex similarity relationships in bipartite graphs. We validate this on two examples. On one hand, we use a deep network to compute accurate and efficient approximations of the marginal distributions for a joint matching problem. This is important to obtain the optimal joint matching hypothesis, for instance when dealing with data association in multi-target tracking (Fortmann, Bar-Shalom, and Scheffe 1980). On the other hand, we use the same architecture to predict the maximum a-posteriori (MAP) estimate of a quadratic assignment problem (QAP), which is known to be NP-hard.

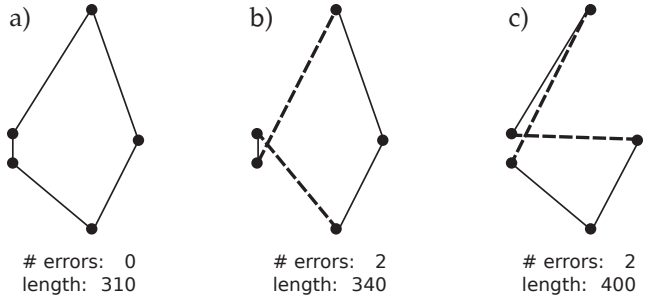

Figure 1: This example illustrates the discrepancy between the loss typically used for training neural networks and the actual objective. Consider a simple TSP and its solution in a). A typical classification loss, as used *e.g.* in (Vinyals, Fortunato, and Jaitly 2015) will rate both b) and c) to be equally bad with 2 misclassified edges (dashed). However, b) is in fact much closer to the optimal solution, which is reflected by the task's objective function, the path's length.

Our second contribution is a simple but effective modification to the classical supervised learning procedure, that takes the problem's objective into account when estimating the network's parameters. Despite their remarkable success, traditional deep learning approaches share one major limitation. The most common strategy is to define the proximity of the network's prediction to the correct solution as a loss function – for instance, the mean squared error (MSE) or the cross-entropy loss for regression and classification problems, respectively. The goal then is to minimise this predefined loss with respect to the network parameters, which results in a non-convex optimisation problem.

We argue that this strategy is suboptimal. The loss is typically chosen to be differentiable and thus convenient to be optimised, but it often does not accurately reflect the network's performance. Considering the well-known travelling salesman problem (TSP), for example, it is not guaranteed that a tour that is 'more similar' to the shortest TSP tour

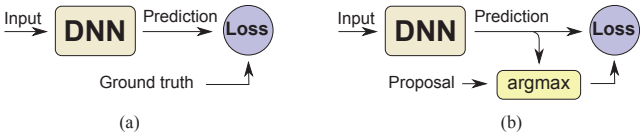

Figure 2: In contrast to the traditional supervised learning architecture *(a)* our proposed extension *(b)* does not require the ground truth for training. Instead, a pre-defined cost function that incorporates the prior knowledge about the problem (*e.g*. argmax), is used to fuse or select from the current prediction and an approximate solution.

w.r.t. the node order, as measured by the log-likelihood classification error (Vinyals, Fortunato, and Jaitly 2015), necessarily yields a shorter length as well. One can easily replace two very short edges from the optimal TSP tour and generate a solution that yields a small loss but results in a highly non-optimal path measured by the tour length. Figure 1 depicts such an example and illustrates how proximity of the prediction to the optimal solution can wrongly rank different candidate solutions in a general case. Moreover, it is a highly demanding task to compute exact solutions to a large number of training instances of an NP-hard problem, making the need to resort to approximate solutions, which are even less reliable for training.

It is therefore advisable to use the problem-specific objective directly to train a deep neural network while taking advantage of suboptimal (or approximate) solutions to a large number of instances of an NP-hard problem. Unfortunately, for many NP-hard problems this objective is non-differentiable and is thus not directly suitable for the standard framework involving gradient descent.

To address this issue, we propose an objective-based learning scheme for deep neural networks to find approximate solutions to NP-hard problems. In order to exploit the full potential of the backpropagation algorithm, we train our network based on a differentiable loss as discussed above. However, we incorporate the task's objective as an indicator to rank the current network's prediction and an approximate precomputed solution (see Fig. 2 *(b)*).

To summarise, we make the following contributions:

1. We present a recurrent neural network capable of learning and predicting bipartite matching in a sequential manner. We explore both, the marginal distributions within a linear objective setting, as well as the MAP estimation of a quadratic assignment. In both cases, obtaining exact solutions becomes intractable for medium-sized problem. Surprisingly, the network is able to learn both tasks from approximate training data only.
2. We introduce a simple, yet effective modification into the traditional supervised learning paradigm. Instead of only relying on an approximate prediction similarity measure (loss), we propose to exploit the true, non-differentiable objective as an additional component to rank solutions and to continually improve on the available approximate 'ground truth'. This enables us to exploit the power of deep architectures and backpropagation, while at the same time allows us to explore large portions of the solution space, without the need for strong supervision.
3. We validate our claims on both synthetic and real data on three applications: Data association for tracking multiple targets, feature point matching and the travelling salesman problem.

Related Work and Background

Deep neural networks (DNNs) have become ubiquitous in our every day lives. A combination of both advances in computing power as well as the sudden increase of available training data, have allowed deep networks to unfold their capabilities on numerous tasks including image classification (Krizhevsky, Sutskever, and Hinton 2012), speech recognition (Graves, Mohamed, and Hinton 2013), machine translation (Sutskever, Vinyals, and Le 2014), and many more. Since the seminal work on the Turing-Completeness of neural networks by Siegelmann and Sontag (1995), many customised network designs have been proposed to address either general or specific tasks. For example, Graves *et al*. (2014) propose an architecture that mimics a Turing machine but is differentiable allowing for standard gradient descent-based training. Somewhat similar, Zaremba *et al*. (2014) show that recurrent neural networks (RNNs) can be employed to learn from data to execute very simple program snippets.

An RNN is used for for human motion forecasting by Fragkiadaki *et al*. (2015), where a representation is implicitly learned by an encoder-recurrent-decoder scheme. This model is generalised by Jain *et al*. (2016) to consider any spatio-temporal conditional random fields (CRFs) that are trainable end-to-end. Another attempt to cast CRFs as RNN was proposed recently by Zheng *et al*. (2015), where the iterative mean field optimisation process is fully integrated into a recurrent neural network.

One of the drawbacks of standard RNNs is that the mapping between the input and the output is implicitly assumed to be known, *i.e*. at each time step, some combination of the hidden state and the input is expected to produce one output. For some problems such as machine translation, this poses a serious limitation because a sentence in the target language generally depends on the entire input sentence. To remedy these shortcomings, (Sutskever, Vinyals, and Le 2014) propose an effective trick to encode the entire source first before starting to make the predictions. While achieving the goal of successful machine translation, this method is not suitable for tasks where the output space is governed by the input, *e.g*. if any solution is a permutation of the input entities. To address that, Vinyals *et al*. (2015) design pointer networks that assemble the output at each step as a collection of all inputs. A beam search is employed to guarantee a feasible solution.

Our work is inspired by the recent application of recurrent networks to tasks like TSP (Vinyals, Fortunato, and Jaitly 2015) or code execution (Zaremba and Sutskever 2014). However, we extend the existing work in several important ways. First, we apply our algorithm on real-world data and show competitive results on standard datasets. Second, we

introduce a simple, yet effective scheme that allows one to bypass the often expensive collection of exact ground truth data. A similar idea has also recently been explored in the context of semantic segmentation (Papandreou et al. 2015). However, in contrast to (Papandreou et al. 2015), our training does not require any strong supervision at all. Garg, Kumar BG, and Reid (2016) have also considered training a deep neural network directly on the problem's objective function, which is in spirit closest to our work. Note, however, that the objective in (Garg, Kumar BG, and Reid 2016) is differentiable, such that standard stochastic gradient descent (SGD) can be directly applied to optimise it. In contrast, we do not make this strong assumption and allow for an arbitrary objective to be integrated into the training.

Recurrent neural networks

Let us briefly recap the functionality of recurrent neural nets (RNNs) for completeness. In its most general form, an RNN is a function that maps a sequence of inputs $\mathbf{x} = [x_1, x_2, \ldots, x_T]$ to a sequence of outputs $\mathbf{y} = [y_1, y_2, \ldots, y_T]$. It does so by maintaining and modifying an internal hidden state $\mathbf{h} = [h_1, h_2, \ldots, h_T]$. More formally, the hidden state at time t is typically computed given the previous hidden state h_{t-1} and the current input x_t as

$$h_t = \mathcal{R}_h \left(W_{ih} x_t + W_{hh} h_{t-1} + b_h \right), \tag{1}$$

where W and b denote the learnable parameters as weights and biases, respectively. $\mathcal{R}$ is a non-linear activation function, usually chosen to be the logistic function or the hyperbolic tangent. The output at time t is extracted from the hidden representation as

$$y_t = \mathcal{R}_o \left(W_{ho} h_t + b_o \right). \tag{2}$$

Note that the same parameters are shared across all time instances.

A very popular extension of the standard RNN model from above is the long short-term memory (LSTM) first presented by Hochreiter and Schmidhuber (1997). It maintains a memory cell c that allows one to capture long-term dependencies. It also implements a gating mechanism that controls what portion of the hidden state should be kept and what should be replaced. This mechanism is modelled by input, output, and forget gates as $i, o, f = \sigma \left[W_{\ldots} x + W_{\ldots} h + b \right]$, where σ is the sigmoid function. In both cases the model complexity can be extended by having multiple layers in each time step.

Our Approach

We will now present our method for sequential prediction of assignments in a bipartite matching problem. We employ the long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997) to make sequential matching predictions, one pair at a time. The input $\mathcal{F}$ is passed through a fully-connected layer and added to the hidden state $\mathbf{h}$. The output is a probability distribution over the number of elements in the second set, obtained by applying a softmax transform. The approach is illustrated in Fig. 3. This architecture is used for predicting both the marginal distribution of the linear assignment and the MAP estimation of the quadratic binary program. We have also considered using the sequence-to-sequence architecture (Sutskever, Vinyals, and Le 2014) to encode the entire input for the matching problem but could not achieve reasonable performance. We believe that this is mainly due to the fact that the input sequences become too long, which is a known limitation of that technique. Concretely, in the case of quadratic assignment with N point pairs, the cost matrix (or feature vector $\mathcal{F}$) contains N^4 elements.

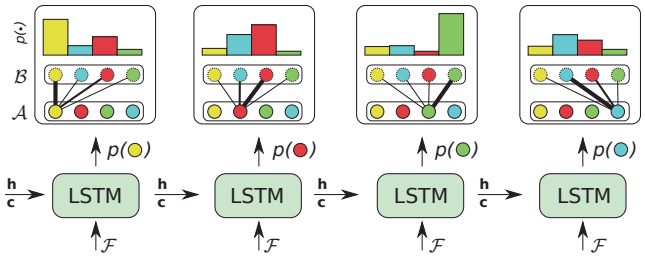

Figure 3: Our LSTM model for solving bipartite matching. At each step, the network outputs a probability distribution for matching one point from set $\mathcal{A}$ to each point in set $\mathcal{B}$.

Objective-based training

As an important technical contribution, we extend our model from above to allow for a so-called *objective-based* learning. To that end, we compute the problem-specific objective at each iteration of gradient descent for both the current network prediction and the available approximate solution to be used as our target. Contrary to traditional loss-based training, we only propagate the gradient if the available solution proposal yields a better objective than our predicted solution (see Fig. 2 for an illustration). We argue that this strategy will steer the training towards higher quality solutions. Moreover, it provides a simple way to generate a more accurate training set during training itself.

Experiments

We demonstrate the efficacy of our proposed method on three examples. First, we show that the marginal distributions of a linear assignment problems can be successfully learned. Second, the MAP solution of a binary quadratic binary program is estimated and used to find feature point correspondences. Third, we show that the objective-based scheme leads to a minor but consistent performance boost on both aforementioned applications as well as the Pointer-Net (Vinyals, Fortunato, and Jaitly 2015) applied on the travelling salesman problem.

Marginalisation of linear assignment for data association

Bipartite matching is a well-known problem in graph theory that arises in many practical applications. Given two sets, the goal here is to find correspondences between elements in the two sets, such that the overall similarity score is maximised, while maintaining a one-to-one relationship. Solving the matching problem is important to find feature

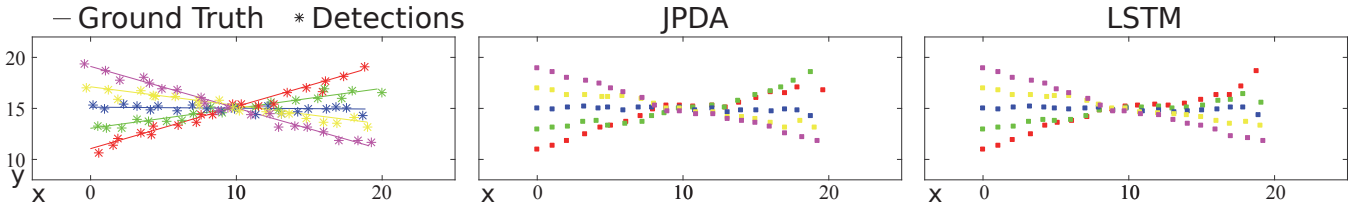

Figure 4: Tracking multiple targets on synthetic data over 20 frames (x, y-coordinates shown). *Left:* Ground truth (solid lines) and noisy detections. *Middle:* Results obtained by JPDA (Fortmann, Bar-Shalom, and Scheffe 1980). *Right:* Our approach (LSTM obj.). Each colour corresponds to a particular target. Note that our LSTM-based data association correctly resolves this crossing case while JPDA switches the red and green trajectories.

point pairs across images for 3D reconstruction (Zamir and Shah 2014) and action recognition (Brendel and Todorovic 2011), to find the same person in different frames of a video for multi-target tracking (Xiong et al. 2013) and person re-identification, or to find similarities between chemical compounds using subgraph isomorphism (Ullmann 1976). The task can be generally formulated as a constrained binary optimisation problem, where the constraints ensure a valid assignment.

In some specific cases such as person re-identification, the problem is addressed as a linear assignment, which can be formulated as a binary linear program

$$X^* = \arg\max_X c^\top X \tag{3}$$

$$\text{s. t. } \forall j : \sum_i \mathcal{X}_{ij} = 1 \quad \wedge \quad \forall i \sum_j \mathcal{X}_{ij} = 1, \tag{4}$$

where c are the (linear) coefficients, $X \in \{0,1\}^{N^2}$ is the binary solution vector and $\mathcal{X} \in \mathbb{R}^{N \times N}$ is the same vector reshaped as a square matrix, where $\mathcal{X}_{ij} = 1$ iff element i is matched to element j. The constraints (4) ensure that $\mathcal{X}$ is an assignment matrix.

It turns out that the globally optimal linear assignment in Eq. (3)-(4) can be found in polynomial time, for instance using the well-known Hungarian (or Munkres) algorithm. However, it was recently shown (Rezatofighi et al. 2016) that matching accuracy can be improved significantly through marginalisation over each entity. More precisely, the marginal probability $\mathbf{p}$ of matching element i to j is obtained by summing over all valid (one-to-one) solutions that contain that match ($\mathcal{X}_{ij} = 1$)

$$\mathbf{p}(\mathcal{X}_{ij}) = \sum_{\{\mathcal{X} \in \Theta | \mathcal{X}_{ij} = 1\}} p(\mathcal{X}), \tag{5}$$

where Θ is the space of all feasible solutions and $p(\mathcal{X}) \propto \prod_{\forall k,l} p(\mathcal{X}_{kl})^{\mathcal{X}_{kl}}$ is the joint probability of one particular assignment hypothesis. Here, $p(\mathcal{X})$ can be calculated by reformulating it as linear assignment problem

$$\log\left(p(\mathcal{X})\right) = \sum_{\forall k,l} c_{kl} \mathcal{X}_{kl} = c^\top X, \tag{6}$$

where $c_{kl} \propto \log\left(p(\mathcal{X}_{kl})\right)$. In contrast to Eq. (3), the marginal probability effectively considers all feasible (not only optimal) solutions simultaneously, which is also the main reason for the high robustness of joint probabilistic data association (JPDA) (Fortmann, Bar-Shalom, and Scheffe 1980) in the context of multi-target tracking.

Method	OSPA-T ↓	ID switches ↓
JPDA	0.41	**0.40**
JPDA_{10}*	0.43	1.10
HA	0.47	1.30
LSTM (loss)	0.42	0.70
LSTM (obj.)	**0.37**	0.60

*Used as training data

Table 1: Tracking performance, averaged over 100 random runs, measured by the OSPA-T error and the number of identity switches. The **best** and second best results for each metric are highlighted in bold red and blue, respectively. Note that only the approximation JPDA_{10} was used for training, but the LSTM is able to capture the structure of the problem, resulting in a slight performance improvement.

Computing the exact marginals is an expensive combinatorial problem because one needs to iterate over exponentially many permutations. Here, we propose to learn to estimate the marginal matching distribution by training a recurrent neural net specifically customised for the task at hand.

As described above, the input matrix[1] $\mathcal{F} \in \mathbb{R}^{N \times N}$ contains the edge weights of the bipartite graph. In our application of tracking multiple targets, these weights are computed as Mahalanobis distances between the estimated target states and all measurements. We use the Kullback-Leibler (KL) divergence as the loss to fit the predicted assignment distribution to the provided (approximate) marginal distribution. The network's size, *i.e.* the dimension of the hidden state vector h is 128 and there is one layer with a dropout probability of 0.1. We use 50 000 training examples and a batch size of 10. The learning rate is set to 0.001 and is decreased by 10% every 1000 iterations.

To validate our method for approximating marginal distributions, we perform experiments on tracking five targets in a simulated setting. The target locations are generated on a 2D plane over 20 frames such that all tracks cross around frame 10 (*cf*. Fig. 4 left). Detections are obtained by adding random Gaussian noise in x and y and permuting

[1] In practice, the matrix is reshaped as a vector.

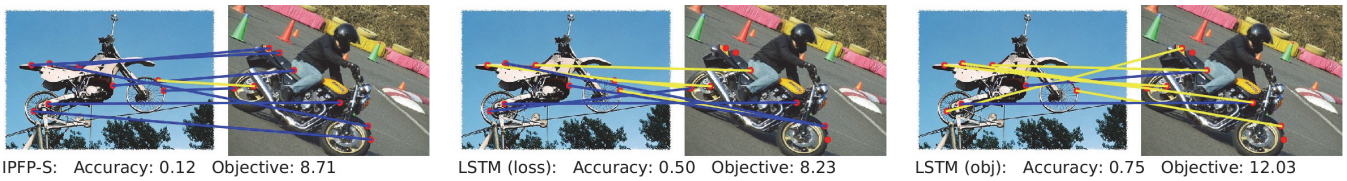

Figure 5: *Left:* One test exemplar of feature point matching using IPFP-S (Leordeanu, Sukthankar, and Hebert 2011); *middle:* our prediction using traditional loss-based training; and *right:* and the objective-based strategy. Yellow and blue lines depict correct and false matches, respectively. Note that with the objective-based training, we achieve better results both in terms of the matching accuracy and *w.r.t.* the objective value than the IPFP-S baseline.

their identities. The tracking is performed using independent Kalman filters that are updated using the weighted data association output of each method. We compare our standard supervised (loss-based) training and our objective-based training approach (LSTM obj.) with three baselines. The joint probabilistic data association (JPDA) (Fortmann, Bar-Shalom, and Scheffe 1980), which uses the exact marginals for each target-to-measurement assignment, its approximation computed from only 10 best association hypotheses (JPDA_{10}) (Rezatofighi et al. 2015), and the MAP solution of the linear assignment from Eq. (3) solved by the Hungarian algorithm (HA). The tracking performance listed in Tab. 1 is measured by two different metrics: OSPA-T location error (Ristic et al. 2011), which combines track accuracy and misassignments, as well as the raw number of ID switches.

Interestingly, the LSTM is able to outperform the JPDA_{10} approximation that was used for training. We believe that this is due to the present structure of the problem at hand that the network is able to capture. Moreover, as expected, the objective-based variant of our training leads to slightly better results for both metrics, indicating the effectiveness of the proposed method.

Quadratic assignment for feature point matching

In feature point matching the task is to find one-to-one point correspondences between two images, and is typically formulated as a quadratic assignment problem (QAP) (Cho, Lee, and Lee 2010; Leordeanu, Sukthankar, and Hebert 2011; Zhou and De la Torre 2012; Zhang et al. 2016). Similar to the linear case above (Eq. (3)), the solution is represented by a binary vector $X \in \{0,1\}^{N^2}$ that should obey the same assignment constraints from Eq. (4). The difference, however, is that here, the objective takes on a quadratic form $X^* = \arg\max_X X^\top Q X$, which makes the problem NP-hard and consequently the computation of the global optimum intractable for real-sized problems. Here, the input matrix $\mathcal{F} \in \mathbb{R}^{N^2 \times N^2}$ carries the pairwise edge terms computed from geometric features as defined in (Leordeanu, Sukthankar, and Hebert 2011). In particular, these features measure the similarity of two interest point pairs *w.r.t.* their distance and orientation defined by their normal vectors.

We use the same architecture as above (*cf*. Fig. 3) to predict one matching pair at a time from the input, but with two minor differences. First, we are interested in computing the optimal solution of the QAP instead of fitting a probability distribution and thus employ the standard log-likelihood loss

Name	Accuracy	Objective	Time [s]
Branch-and-cut	**0.90**	**10.99**	0.007
IPFP-S	0.66	10.11	0.009
IPFP-S (best of 10)	0.70	10.47	0.056
LSTM (loss)	0.68	9.54	**0.004**
LSTM (obj.)	0.76	10.52	0.004

Table 2: Bipartite matching with quadratic costs, averaged over 10 samples from the motorbike test set. The **best** and second best results for each metric are highlighted in bold red and blue, respectively.

to measure the number of mismatches. Second, for this task, we use a 2-layered RNN of size 64. Reducing the network size for a more complex problem may seem counterintuitive, however we found that this regularisation is necessary due to the increased number of parameters induced by the larger input feature $\mathcal{F}$. The remainder of the settings are equivalent to the network described in the previous section.

For this experiment, we employ the public car and motorbikes dataset of Leordeanu *et al*. (2011), consisting of 30 pairs of car images and 20 pairs of motorbikes images with manually annotated point correspondences.[2] We train our network to match eight point pairs. To generate the training and validation data, we randomly sample the points from car images for each training instance and obtain 100 approximate solutions using the Integer Projected Fixed Point solver (IPFP) (Leordeanu, Sukthankar, and Hebert 2011) combined with a naive exclusion strategy proposed in (Rezatofighi et al. 2016). The motorbikes set is used exclusively for testing. We do not add any outliers in our experiments. Note that we demonstrate the efficacy of the proposed method on this rather small-sized problem, which allows us to validate the solution with respect to the global optimum.

Tab. 2 lists quantitative results of our method, compared to several baselines. Gurobi's QBP solver (Gurobi Optimization 2015) is able to find the optimal solution for this problem size. Note, however, that this is not the case for larger instances. The IPFP-S solver (Leordeanu, Sukthankar, and Hebert 2011) is efficient, but often leads to suboptimal matching solutions. Choosing the best out of ten computed IPFP solutions improves the result, but necessarily requires a large computational overhead. Out LSTM-based method can match the approximate solver in terms of accuracy and

[2]Note that these annotations are *not* used for training.

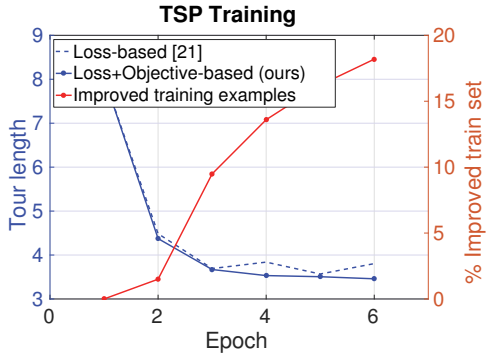

Figure 6: Blue curves show the decrease of the predicted tour length on the training set for the traditional loss-based training (Vinyals, Fortunato, and Jaitly 2015) (dashed) and our proposed approach. The red graph depicts the percentage of predicted solutions that yield a lower length than the provided approximate solutions.

Method	Nearest neighbours			Ptr-Net	Ptr-Net
	Best 20	Avg 20	Worst 20	(orig.)	(ours)
Tour length	3.09	3.43	3.87	3.80	3.45

Table 3: TSP test results after 6 epochs, averaged over 1000 20-node problems.

is at the same time twice as fast as one single IPFP iteration. When using the objective to select the optimal solution proposals during training (LSTM obj.), we are able to improve the results even further, while maintaining the same efficiency during inference. Fig. 5 shows qualitative results.

Travelling salesman problem

To further demonstrate the benefits of the proposed objective-based learning method over traditional supervised learning with approximate ground truth, we explore the well-known travelling salesman problem (TSP). To that end, we use a publicly available implementation[3] of Pointer Networks (Vinyals, Fortunato, and Jaitly 2015) as our baseline architecture, which is slightly different from the LSTM for bipartite matching, and experimentally validate that our objective-based learning generalises to a variety of network architectures and NP-hard problems.

Unlike the work of Pointer-Net, which shows the basic functionality of the approach on randomly generated TSP instances, we focus on solving the travelling salesman problem on a real map. We use the map of Qatar (Lelonek 2013) with 194 different locations as our playing field for learning to solve the TSP with 20 nodes. Even on such relatively small map one can encounter nearly 10^{27} unique 20-node travelling salesman problems. We randomly sample only 10K subsets from the large pool and aim to learn a network without strong supervision, which can in turn be used to accurately solve any new TSP on the same map.

[3]https://github.com/vshallc/PtrNets

Although a vast number of approximate TSP algorithms exist, without loss of generality we use a simple ensemble of 20 nearest neighbour (NN) solutions (one starting from each node) as our approximate solver for training. In other words, for each training instance we choose the best-of-20 NN solutions as our initial ground truth.

In our experiment, we train the Pointer-Net on 10K training examples with the standard log-likelihood loss, a batch size of 1, and learning rate of 1 using stochastic gradient decent. We then repeat the exact same experiment with the exception that instead of blindly relying on the gradient of the log-likelihood loss function, before every gradient upgrade we project the predicted tour onto the valid solution space via beam search and compare the tour length of the prediction to that of the nearest neighbour algorithm. Finally, we only back-propagate the gradients for those instances whose prediction yields a longer tour, as described earlier. Fig. 6 shows that including the objective-based gradient replacement yields a faster and more consistent convergence with marginally better results, compared to Pointer-Net. What is even more striking is that already after very few epochs, the predictions outperform the provided 'ground truth' in nearly 20% of all training instances. As Fig. 6 (red) illustrates, this effect of bootstrapping is rather strong in the beginning and flattens out as the model becomes more and more accurate. As expected, better training examples continually lead the model towards better predictions, averaged over the dataset. This once again supports our claim that the objective-based training is an effective way to generate more accurate training data that can in turn be used to further refine both the network and the training data iteratively.

Discussion and Limitations

We showed that the proposed approach gives reasonable approximations to complex combinatorial problems. Nevertheless, we would like to point out some existing limitations and to discuss important aspects that should not be ignored.

One of the most obvious drawbacks of the presented method is the relatively long training procedure that is necessary, which is encountered in almost all existing techniques that deal with deep learning and such huge numbers of free parameters. Even though this is a completely offline process and does not pose an issue for most practical applications, it should not be left out of sight. Another important limitation factor is the fixed input and output size of the matching problem. Note that we have considered using the Pointer-Net architecture to allow for arbitrary sized problems, however, this is non-trivial because we are dealing with a structured and not only sequential input. The main goal of this work is to provide first steps towards data-driven, objective-based learning of complex algorithms. Although we believe that dealing with arbitrary sized graphs is an important issue to be addressed, it lies outside the scope of this paper and we leave it for future work.

Conclusion and Outlook

We have presented a method for learning to solve complex combinatorial problems. The proposed architecture exploits

the sequential process of RNNs and allows for a step-by-step prediction of the final solution. Our experiments on multiple applications show that this learning-based approach results in very good approximations of the globally optimal solution, while taking only a fraction of the computational resources. Furthermore, we showed that by incorporating the problem specific objective, one is able to bypass the expensive step of generating exact ground truth, rendering our approach "less supervised". In future, we plan to further investigate the possibility to extend our method to arbitrary sized problems, using either an input embedding strategy, or exploring convolutional techniques for general graphs. It is also possible to explore the application of the presented approach to other NP-hard problems.

Acknowledgments. This work was supported by ARC Linkage Project LP130100154, ARC Laureate Fellowship FL130100102 and the ARC Centre of Excellence for Robotic Vision CE140100016.

References

Brendel, W., and Todorovic, S. 2011. Learning spatiotemporal graphs of human activities. In *ICCV*.

Cho, M.; Lee, J.; and Lee, K. M. 2010. Reweighted random walks for graph matching. In *ECCV*, 492–505.

Fortmann, T. E.; Bar-Shalom, Y.; and Scheffe, M. 1980. Multi-target tracking using joint probabilistic data association. In *19th IEEE Conference on Decision and Control*, volume 19, 807–812.

Fragkiadaki, K.; Levine, S.; Felsen, P.; and Malik, J. 2015. Recurrent network models for human dynamics. In *ICCV*.

Garg, R.; Kumar BG, V.; and Reid, I. 2016. Unsupervised CNN for single view depth estimation: Geometry to the rescue. *arXiv:1603.04992 [cs]*. arXiv: 1603.04992.

Graves, A.; Mohamed, A.-r.; and Hinton, G. E. 2013. Speech recognition with deep recurrent neural networks. In *ICASSP 2013*, 6645–6649.

Graves, A.; Wayne, G.; and Danihelka, I. 2014. Neural turing machines. *CoRR* abs/1410.5401.

Gurobi Optimization, I. 2015. *Gurobi Optimizer Reference Manual*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Comput.* 9(8):17351780.

Jain, A.; Zamir, A. R.; Savarese, S.; and Saxena, A. 2016. Structural-RNN: Deep learning on spatio-temporal graphs. In *CVPR*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*, 1097–1105.

Lelonek, K. 2013. Traveling salesman problem solver in coffeescript alogorithm. https://github.com/KamilLelonek/Traveling-Salesman-Problem/. Accessed: 2016-08-30.

Leordeanu, M.; Sukthankar, R.; and Hebert, M. 2011. Unsupervised learning for graph matching. *IJCV* 96(1):28–45.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533.

Papandreou, G.; Chen, L.-C.; Murphy, K. P.; and Yuille, A. L. 2015. Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *ICCV*.

Rezatofighi, S. H.; Milan, A.; Zhang, Z.; Shi, Q.; Dick, A.; and Reid, I. 2015. Joint probabilistic data association revisited. In *ICCV*.

Rezatofighi, S. H.; Milan, A.; Zhang, Z.; Shi, Q.; Dick, A.; and Reid, I. 2016. Joint probabilistic matching using m-best solutions. In *CVPR*.

Ristic, B.; Vo, B. N.; Clark, D.; and Vo, B. T. 2011. A metric for performance evaluation of multi-target tracking algorithms. *IEEE Transactions on Signal Processing* 59(7):3452–3457.

Siegelmann, H. T., and Sontag, E. D. 1995. On the computational power of neural nets. *Journal of Computer and System Sciences* 50(1):132150.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Driessche, G. v. d.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; Dieleman, S.; Grewe, D.; Nham, J.; Kalchbrenner, N.; Sutskever, I.; Lillicrap, T.; Leach, M.; Kavukcuoglu, K.; Graepel, T.; and Hassabis, D. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529:484–503.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Ullmann, J. R. 1976. An algorithm for subgraph isomorphism. *J. ACM* 23(1):3142.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. In *NIPS*, 2692–2700.

Xiong, H.; Zheng, D.; Zhu, Q.; Wang, B.; and Zheng, Y. 2013. A structured learning-based graph matching method for tracking dynamic multiple objects. *IEEE Transactions on Circuits and Systems for Video Technology* 23(3):534–548.

Zamir, A., and Shah, M. 2014. Image geo-localization based on multiple nearest neighbor feature matching using generalized graphs. *IEEE TPAMI* 36(8):1546–1558.

Zaremba, W., and Sutskever, I. 2014. Learning to execute. *arXiv:1410.4615 [cs]*. arXiv: 1410.4615.

Zhang, Z.; Shi, Q.; McAuley, J.; Wei, W.; Zhang, Y.; and van den Hengel, A. 2016. Pairwise matching through max-weight bipartite belief propagation. In *CVPR*.

Zheng, S.; Jayasumana, S.; Romera-Paredes, B.; Vineet, V.; Su, Z.; Du, D.; Huang, C.; and Torr, P. 2015. Conditional random fields as recurrent neural networks. In *ICCV*.

Zhou, F., and De la Torre, F. 2012. Factorized graph matching. In *CVPR*.