Approximate Structured Diffusion for Sequence Labelling

Anonymous ACL submission

Abstract

Sequence labelling, a core task of Natural Language Processing (NLP), consists in assigning each token of an input sentence a label. From a Machine Learning point of view, sequence labelling is often cast as a Linear-Chain Conditional Random Field (CRF) parametrised by a neural network. While this approach gives good empirical results, CRFs assume a finite decision span (*e.g.* label bigrams) which limits their expressivity and hurts performance when long-range dependencies are required.

We show we can leverage diffusion to train a CRF conditioned on an entire label sequence, with the caveat that the condition is on a *noisy* version of labels. We show experimentally that this method, in conjunction with Mean-Field inference, improves label accuracy with a 16.5% error reduction while remaining faster than traditional CRF decoding.[1]

1 Introduction

Sequence labelling, a fundamental task in NLP, consists in assigning a tag to each token of an input sentence. It is the foundation of a variety of NLP applications, such as POS tagging, named-entity recognition or parsing. Modern approaches to this task are based on CRFs parametrised by neural networks (Lafferty et al., 2001; Zheng et al., 2015). While structured models such as CRFs consider interactions between labels, tractability impose some restrictions. Thus many proposed models are limited to bigrams, *i.e.* correlations of adjacent labels.

Recently, diffusion has been applied to language modelling to effectively condition generation on unbounded contexts (Hoogeboom et al., 2021; Austin et al., 2021; Sahoo et al., 2024). In practice, these models train denoisers to predict *independently* each token from noisy versions of the clean output.

In this work, we bridge these two concepts, *i.e.* structured prediction and discrete diffusion, for sequence labelling. We define a CRF conditioning the predicted label sequence not only on the input sentence but also on a noisy label sequence. This helps the model consider unbounded label interactions while remaining able to enforce preferences on predicted adjacent labels. Decoding with diffusion models requires iterative sampling to refine predictions from random noise. Since sampling CRF distributions is costly, with a complexity linear in the input size, we speed up decoding and training by approximating them with Mean-Field.

We evaluate on POS tagging and show that this model scales up better than baseline CRFs, with both the unigram diffusion model and the addition of either a CRF denoiser or its Mean-Field approximation achieving superior performance.

2 Model

2.1 Standard Sequence Labelling Model

Given a sentence $\boldsymbol{s} = s_1 \dots s_n$, with s_i the i^{th} word, labelling produces a sequence $\boldsymbol{y} = y_1 \dots y_n$, with $y_i \in \mathcal{L}$ the label for s_i. More precisely, standard sequence labelling models define a parametrised probability distribution $p_\theta(\boldsymbol{y}|\boldsymbol{s})$ so labelling amounts to returning the mode $\widehat{\boldsymbol{y}} = \operatorname{argmax}_{\boldsymbol{y}} p_\theta(\boldsymbol{y}|\boldsymbol{s})$ and learning parameters θ is cast as Maximum Likelihood Estimation. These distributions are usually written as energy models $p_\theta(\boldsymbol{y}|\boldsymbol{s}) \propto \exp f(\boldsymbol{x}, \boldsymbol{y}; \theta)$, computed by a neural network implementing f, *i.e.* parameters θ are the parameters of f. The decomposition of f over sequences is crucial for efficiency.

Unigram Models sum unary potential over the sequence $f(\boldsymbol{s}, \boldsymbol{y}; \theta) = \sum_{i=1}^{n} f(\boldsymbol{s}, y_i; \theta)$. Typically this is implemented as a Transformer (Vaswani et al., 2017) whose output vector at position i feeds a MLP computing f for all labels at this position. As a consequence of the decomposition of f, p_θ

[1] Our code will be made available publicly upon acceptation.

is factorized, *i.e.* $p_\theta(\boldsymbol{y}|\boldsymbol{s}) = \prod_{i=1}^{n} p_\theta(y_i|\boldsymbol{s})$. Labelling and training are efficient but the independence between predictions impairs the expressivity required to model fine-grained label interactions.

Bigram Models sum unary and binary[2] potentials over adjacent positions, $f(\boldsymbol{s}, \boldsymbol{y}; \theta) = \sum_{i=1}^{n} f_1(\boldsymbol{s}, y_i; \theta) + \sum_{i=1}^{n-1} f_2(\boldsymbol{s}, y_i, y_{i+1}; \theta)$. Transformer output are fed to f_1 like in the unigram case. For f_2, we need to compute a transition matrix for all labels y_i, y_{i+1}. This can be either implemented as a position-independent matrix, or as a MLP computing at each position the transition weights, fed up by a Transformer's output. Labelling and training can be performed in linear time w.r.t. sentence's length, as can be computed marginal probabilities, with Viterbi (Forney, 1973) and Forward/Backward (Rabiner, 1989) algorithms. Bigram CRFs are difficult to parallelise but approximations such as Mean-Field (Wang et al., 2020) or Mean-Regularisation (Corro et al., 2025) recover the position-wise independent computation, and thus the efficiency, of unigram models.

2.2 Discrete Diffusion Models for Labelling

We follow diffusion language models (Hoogeboom et al., 2021; Austin et al., 2021) to define labelling as generation of labels given words.

Forward Diffusion. A sequence of tags $\boldsymbol{y}^0$ is altered by a forward diffusion process q consisting of T steps $\mathbf{y}^1 \dots \mathbf{y}^T$ to eventually obtain a random sequence[3] $\boldsymbol{y}^T$. Generating such sequences is a Markovian process $q(\mathbf{y}^1 \dots \mathbf{y}^T|\mathbf{y}^0) = \prod_{t=1}^{T} q_t(\mathbf{y}^t|\mathbf{y}^{t-1})$ with independent noise at each position i: $q_t(\boldsymbol{y}^t|\boldsymbol{y}^{t-1}) = \prod_{i=1}^{n} q_t(y_i^t|y_i^{t-1})$.

Noise distributions are parametrised by a corruption ratio β_t following a predefined schedule:[4]

$$q_t(y_i^{t+1}|y_i^t) = \begin{cases} \beta_t + \frac{\beta_t}{|\mathcal{L}|} \text{ if } y_i^{t+1} = y_i^t \\ \frac{1-\beta_t}{|\mathcal{L}|} \text{ otherwise} \end{cases}$$

The parameters of these conditional distributions $q_t(\cdot|\cdot)$ for timestep t can be encoded as a matrix Q_t. We can also precompute consecutive applications of t diffusion steps $q_{0|t}(y_i^t|y_i^0) = \sum_{y_i^{t-1}} q_t(y_i^t|y_i^{t-1}) q_{0|t-1}(y_i^{t-1}|y_i^0)$.

Denoising. Our model, following Hoogeboom et al. (2021); Austin et al. (2021) for language models, produces a parameterised distribution on label sequences from a random sequence by reversing the diffusion process. With a slight abuse of notation we also denote this distribution as p_θ. This model can assign a probability to less and less noisy sequences, also a markovian process : $p_\theta(\mathbf{y}^0\mathbf{y}^1 \dots \mathbf{y}^T|\boldsymbol{s}) = p(\mathbf{y}^T) \prod_{i=t}^{T} p_\theta(\mathbf{y}^{t-1}|\mathbf{y}^t, \mathbf{s})$, where the prior $p(\boldsymbol{y}^T)$ is the uniform distribution. We drop the condition on $\boldsymbol{s}$ in notations.

Denoiser p_θ is implemented by a neural network presented in §2.3. The same network is used for all t: to add time information, we feed the neural network with a learned representation of t. We follow the widely adopted architecture of Ho et al. (2020) and describe a single denoising step from t as full denoising followed by $(t-1)$ forward steps:

$$\begin{aligned} p_\theta(\boldsymbol{y}^{t-1}|\boldsymbol{y}^t) &= \sum_{y^0} p_\theta(\boldsymbol{y}^0|\boldsymbol{y}^t) q(\boldsymbol{y}^{t-1}|\boldsymbol{y}^t, \boldsymbol{y}^0) \\ &= \mathbb{E}_{\boldsymbol{y}^0 \sim p_\theta(\cdot|\boldsymbol{y}^t)} \left[q(\boldsymbol{y}^{t-1}|\boldsymbol{y}^t, \boldsymbol{y}^0) \right] \\ &\approx q(\boldsymbol{y}^{t-1}|\boldsymbol{y}^t, \widehat{\boldsymbol{y}^0}) \\ &\text{with } \widehat{\boldsymbol{y}^0} = \mathbb{E}_{\boldsymbol{y}^0 \sim p_\theta(\cdot|\boldsymbol{y}^t)} [\boldsymbol{y}^0]. \end{aligned}$$

A denoising step can thus be modelled as sampling from (i) the so-called posterior distribution with (ii) the clean sequence $\boldsymbol{y}_0$ replaced by an expected sequence $\widehat{\boldsymbol{y}^0}$. In practice, addressing (i) requires computing the posterior distribution, expressed with three tractable distributions, from Bayes' theorem and Markovian assumption :

$$\begin{aligned} q(\boldsymbol{y}^{t-1}|\boldsymbol{y}^t, \boldsymbol{y}^0) &= \frac{q(\boldsymbol{y}^{t-1}, \boldsymbol{y}^t|\boldsymbol{y}^0)}{q(\boldsymbol{y}^t|\boldsymbol{y}^0)} \\ &= \frac{q(\boldsymbol{y}^t|\boldsymbol{y}^{t-1}, \boldsymbol{y}^0) q(\boldsymbol{y}^{t-1}|\boldsymbol{y}^0)}{q(\boldsymbol{y}^t|\boldsymbol{y}^0)} \\ &= \frac{q_t(\boldsymbol{y}^t|\boldsymbol{y}^{t-1}) q(\boldsymbol{y}^{t-1}|\boldsymbol{y}^0)}{q(\boldsymbol{y}^t|\boldsymbol{y}^0)}. \end{aligned}$$

While explanations above indicated that denoising is performed step by step from t to $t-1$, we can rewrite it to perform several steps at once, from t to $t-k$. This may impact the quality of the generated sequence since the denoiser is called less, and has thus less opportunities to rely on the input sequence. In our experiments we use a *halving* strategy and go from step t to step $\lfloor \frac{t}{2} \rfloor$, starting with step T until we reach 0, so the number of calls to the denoiser is logarithmic in the number of diffusion steps.

Structured Denoising We can adapt the previous decoding method to the case where the denoiser p_θ is implemented by a CRF or an approximation. Remember that the denoiser's role is to gen-

[2] These models can be extended to n-ary potentials.
[3] All sequences of size $|\boldsymbol{y}^0|$ are uniformly probable.
[4] We only consider cosine (Hoogeboom et al., 2021).

erate $\widehat{\boldsymbol{y}^0} = \mathbb{E}_{\boldsymbol{y}^0 \sim p_\theta(\cdot|\boldsymbol{y}^t)}[\boldsymbol{y}^0]$. In the case of linear-chain CRFs we can compute marginals in $O(n)$ time complexity, either by running the forward-backward algorithm (Rabiner, 1989) or backpropagating through the log-partition (Eisner, 2016). Unfortunately, this approach is intractable in our context. Because of the limited parallelizability of the Viterbi algorithm and its variants, training becomes too slow. The linear space complexity of these methods also burdens training with memory consumption. Moreover the denoiser must be called multiple times during decoding.

Instead, we can approximate the CRF distribution with Mean Regularisation (Corro et al., 2025) or find the closest factorized distribution with Mean-Field (Wang et al., 2020). We experiment with the latter and show we can exploit structures with diffusion models while remaining efficient.

Training is performed by maximizing likelihood with the denoiser synced to the diffusion model at each timestep. More precisely we optimise a variational lower bound of the log-likelihood :

$$\begin{aligned}
&\log p_\theta(\boldsymbol{y}^0) = \log \sum_{y^1 \dots y^T \sim q(\cdot|y^0)} p_\theta(\boldsymbol{y}^0, \boldsymbol{y}^1, \dots, \boldsymbol{y}^T) \\
&\geq \mathop{\mathbb{E}}_{\boldsymbol{y}^1 \sim q_{0|1}(\cdot|\boldsymbol{y}^0)} \Big[\log p_\theta(\boldsymbol{y}^0 \mid \boldsymbol{y}^1) \Big] \\
&\quad - \sum_{t=2}^{T} \mathop{\mathbb{E}}_{\boldsymbol{y}^t \sim q_{0|t}(\cdot|\boldsymbol{y}^0)} \left[D_{\mathrm{KL}} \left(\begin{matrix} q(\boldsymbol{y}^{t-1} \mid \boldsymbol{y}^t, \boldsymbol{y}^0) \\ || \; p_\theta(\boldsymbol{y}^{t-1} \mid \boldsymbol{y}^t) \end{matrix} \right) \right] \\
&\quad - D_{\mathrm{KL}} \Big(q_{0|T}(\boldsymbol{y}^T \mid \boldsymbol{y}^0) \,\Big\|\, p(\boldsymbol{y}^T) \Big),
\end{aligned}$$

where D_{KL} is the Kullback-Liebler divergence between the two distributions. The last term can be ignored since, by definition, the two distributions are uniform and their divergence is thus zero. We train our model by uniformly sampling a t between 1 and T, and then a sequence $\boldsymbol{y}^t \sim q_{0|t}(\cdot|\boldsymbol{y}^0)$, which simulates t diffusion steps. If $t = 1$, we only consider the first term; si $t \geq 2$ we only consider the mean in the second term based on this t, which brings us back to optimizing a single KL divergence.

We notice that once sampled $\boldsymbol{y}^t$, the first term is just a log-likelihood. For the second term, we get:

$$\begin{aligned}
&D_{KL}[q(\boldsymbol{y}^{t-1}|\boldsymbol{y}^t, \boldsymbol{y}^0)||p_\theta(\boldsymbol{y}^{t-1}|\boldsymbol{y}^t)] \\
=&D_{KL}[q(\boldsymbol{y}^{t-1}|\boldsymbol{y}^t, \boldsymbol{y}^0)||q(\boldsymbol{y}^{t-1}|\boldsymbol{y}^t, \widehat{\boldsymbol{y}^0})] + C,
\end{aligned}$$

i.e. we seek to match posteriors conditioned on respectively gold sequences and the denoised ones.

Finally, we add the denoising loss (Austin et al., 2021) as it helps training stability and convergence.

2.3 Neural Architecture

The neural architecture depicted in Fig.1 implements potential functions, f for unigrams or f_1, f_2 for bigrams, as defined in §2.1, which are then summed to define probability distributions. All models start converting words $s_1, \dots, s_n$ to non-contextual representations with a look-up table and a charLSTM (Lample et al., 2016). These are contextualized with Transformers (Vaswani et al., 2017) to obtain vectors $e_1, \dots, e_n$. Alternatively, pretrained embeddings can be used. The unigram model is parametrised by n vectors l_i of $|\mathcal{L}|$ scores computed by a MLP from e_i. The bigram models adds $|\mathcal{L}| \times |\mathcal{L}|$ scores which represent label transitions from one position to the next. These are computed for each position by a MLP.

Our diffusion models implement the denoiser with Diffusion Transformer blocks (Peebles and Xie, 2022). It takes as input a sequence of label embeddings from a trainable look-up table corresponding to noisy labels and, as context for normalization, the concatenation of the contextual word embeddings and a trainable embedding of the timestep. After these blocks a final MLP converts position vectors to unigram, and possibly bigram, scores.

3 Experiments

Data We experiment on 4 datasets from Universal Dependencies v2.15 (Zeman et al., 2024) namely EN-EWT, DE-GSD, FR-GSD, NL-LassySmall for English, German, French, and Dutch. We use standard splits, evaluate accuracy with punctuation and average results over 8 random seeds.

We compare the 3 baseline models, unigram, Mean-Field and CRF with our diffusion models, the unstructured and structured models, Diffusion-Uni and Diffusion-MF respectively. Additionally, we report the best POS taggging results from (Corro et al., 2025). The hyper-parameter setups for each model are described in F, while additional results, with a different encoder, features and a baseline for Diff-CRF (the non approximated structured diffusion model) in B, an ablation study in C, and the models' speeds in E.

Our results in Table 1 show that structured diffusion improves performances in nearly all datasets, and get an average 16.54% error reduction between the best model without diffusion (`CRF`) and the model with structured diffusion (`Diffusion-MF`). We find that the diffusion approach allows models to scale better with more parameters, surpass-

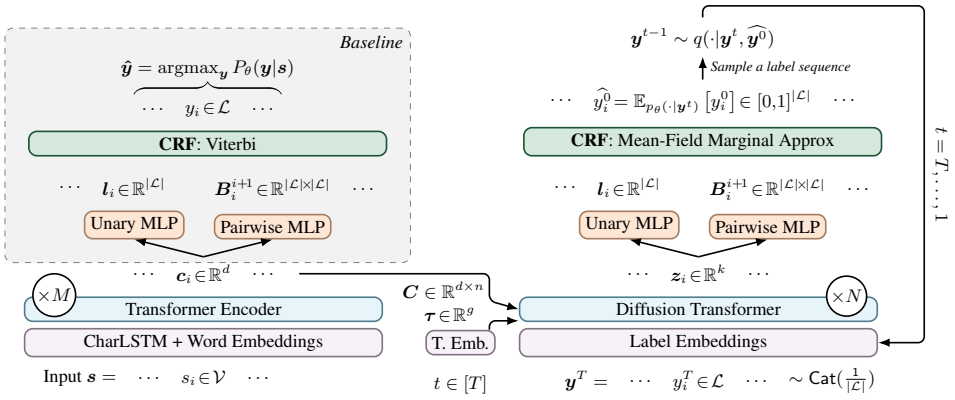

Figure 1: From word embeddings, instead of directly predicting distribution parameters (baseline, left) we use them with timesteps as context for a diffusion transformer fed with sequences of noisy label embeddings (ours, right).

	English	German	French	Dutch	AVG
(Corro et al., 2025)	91.9	94.4	96.5	94.7	94.37
`Unigram`	91.48	92.38	94.58	91.94	92.60
`Mean-Field`	93.51	93.88	96.02	93.84	94.31
`CRF`	94.02	94.11	96.60	94.00	94.68
`Diffusion-Uni`	95.01	94.73	97.33	94.73	95.45
`Diffusion-MF`	**95.06**	**94.85**	**97.39**	**94.93**	**95.56**

Table 1: Test results for UD 2.15.

ing the 3 baseline with an equal parameter count, this is shown in Table C in Appendix. Moreover in this setting, while the main source of performance increase is the use of diffusion, we can see that the additional structure given by the CRF (`Diffusion-MF`) improves performance over the simple diffusion process (`Diffusion-Uni`).

4 Related Work

Structured discrete diffusion. Discrete diffusion either corrupts labels directly (Hoogeboom et al., 2021; Austin et al., 2021) or relaxes tokens for Gaussian diffusion and discretise at decode time (*e.g.* Ho et al., 2020; Li et al., 2022; Peebles and Xie, 2022); related variants use iterative masking (Chang et al., 2022), bit encodings (Chen et al., 2022), or VQ codebooks (Gu et al., 2021). For *structured* NLP, diffusion has been applied to span-level NER (Shen et al., 2023), token-level labelling via bit-relaxed sequences (Huang et al., 2023), and non-autoregressive constrained generation (Gong et al., 2022); continuous-time categorical formulations connect to jump processes and score matching (Sun et al., 2023). Adjacent works insert CRFs around diffusion, *e.g. continuous* CRF for latent diffusion (Ranasinghe et al., 2024) or diffusion-enhanced BiLSTM-CRF for NER (Qiu et al., 2025). Unlike previous works, we use a *CRF denoiser inside the loop*, yielding (i) guaranteed normalization, (ii) global context via the evolving noisy sequence, and (iii) parallelizable mean-field updates.

Structured prediction with neural CRFs. Neural CRF models remain strong for labelling (Lafferty et al., 2001); differentiable inference via unrolled mean-field (CRF-as-RNN) and parallel accelerations improve efficiency (Zheng et al., 2015; Wang et al., 2020; Corro et al., 2025), alongside amortized perspectives (Stoyanov and Eisner, 2011; Domke, 2012; Hershey et al., 2014) and classical variational analyses (Wainwright and Jordan, 2008; Yedidia et al., 2005; Murphy et al., 1999). Rather than single-shot CRF decoding, we perform *repeated, globally informed* CRF diffusion-driven updates, reconciling long-range evidence with local constraints beyond purely accelerated CRF decoders (Wang et al., 2020; Corro et al., 2025). Our model is then closer to (Jayasumana et al., 2024) but with the CRF inside a diffusion denoiser instead of generative Transformers.

5 Conclusion

We presented a novel approach to sequence labelling, with application to POS tagging, based on structured prediction and discrete diffusion to better predict tag distributions. Our model improves the tagging metric and increases parameter scaling, surpassing the baseline and performing even better as the number of parameters grows while keeping a manageable time complexity thanks to the Mean-Field approximation of the CRF distribution. Our approach could also be applied in other tagging tasks in NLP, such as NER, or word segmentation.

Limitations

We showed that the presented method can scale, *i.e*. the more parameters the better accuracy is, as opposed to prior methods which tend to overfit when the number of parameters grow. However, this increase come at the expense of memory consumption and compute time. In other words, our models require more energy to be run at their full potential.

In order to improve parallelization, we resort to the parallel version of Mean-Field for which convergence is not guaranteed. Although we didn't see pathological divergence in practice, we note that the method recently developed by Corro et al. (2025) could be used as a drop-in replacement for parallel Mean-Field with convergence guarantees.

Ethical Considerations

We believe that our work does not raise ethical concerns. We present a novel architecture for sequence labelling based on diffusion and structured prediction and we test it on standard, publicly available data.

We acknowledge the environmental impact of the energy cost of training neural models.

References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pages 17981–17993. Curran Associates, Inc.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. 2022. Maskgit: Masked generative image transformer. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11305–11315.

Ting Chen, Ruixiang Zhang, and Geoffrey E. Hinton. 2022. Analog bits: Generating discrete data using diffusion models with self-conditioning. *ArXiv*, abs/2208.04202.

Caio Corro, Mathieu Lacroix, and Joseph Le Roux. 2025. Bregman conditional random fields: Sequence labeling with parallelizable inference algorithms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 29557–29574, Vienna, Austria. Association for Computational Linguistics.

Justin Domke. 2012. Generic methods for optimization-based modeling. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pages 318–326, La Palma, Canary Islands. PMLR.

Jason Eisner. 2016. Inside-outside and forward-backward algorithms are just backprop (tutorial paper). In *Proceedings of the Workshop on Structured Prediction for NLP*, pages 1–17, Austin, TX. Association for Computational Linguistics.

George David Forney. 1973. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2022. Diffuseq: Sequence to sequence text generation with diffusion models. *ArXiv*, abs/2210.08933.

Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. 2021. Vector quantized diffusion model for text-to-image synthesis. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10686–10696.

John R. Hershey, Jonathan Le Roux, and Felix Weninger. 2014. Deep unfolding: Model-based inspiration of novel deep architectures. *ArXiv*, abs/1409.2574.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems*, volume 34, pages 12454–12465. Curran Associates, Inc.

Ziyang Huang, Pengfei Cao, Jun Zhao, and Kang Liu. 2023. Diffusionsl: Sequence labeling via tag diffusion process. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. To appear.

Sadeep Jayasumana, Daniel Glasner, Srikumar Ramalingam, Andreas Veit, Ayan Chakrabarti, and Sanjiv Kumar. 2024. Markovgen: Structured prediction for efficient text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9316–9325.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. 2022. Diffusion-lm improves controllable text generation. *ArXiv*, abs/2205.14217.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. 1999. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 467–475. Morgan Kaufmann.

William Peebles and Saining Xie. 2022. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Yunfei Qiu, Libo Dong, Wenwen Zhang, Haoran Xing, and Junwei Huang. 2025. A diffusion enhanced crf and bilstm framework for accurate entity recognition. *Scientific Reports*, 15:19670.

Lawrence R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285.

Kanchana Ranasinghe, Sadeep Jayasumana, Andreas Veit, Ayan Chakrabarti, Daniel Glasner, Michael S Ryoo, Srikumar Ramalingam, and Sanjiv Kumar. 2024. Latentcrf: Continuous crf for efficient latent diffusion. *Preprint*, arXiv:2412.18596.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Diffusionner: Boundary diffusion for named entity recognition. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3875–3890. Association for Computational Linguistics.

Veselin Stoyanov and Jason Eisner. 2011. Learning cost-aware, loss-aware approximate inference policies for probabilistic graphical models. In *NIPS Workshop on Structured Prediction and Approximate Inference*. Workshop on Advances in Structured Prediction.

Haoran Sun, Lijun Yu, Bo Dai, Dale Schuurmans, and Hanjun Dai. 2023. Score-based continuous-time discrete diffusion models. In *The Eleventh International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Martin J. Wainwright and Michael I. Jordan. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2020. AIN: Fast and accurate sequence labeling with approximate inference network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6019–6026, Online. Association for Computational Linguistics.

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. 2005. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312.

Daniel Zeman, Joakim Nivre, Mitchell Abrams, Elia Ackermann, Noëmi Aepli, Hamid Aghaei, Željko Agić, Amir Ahmadi, Lars Ahrenberg, Chika Kennedy Ajede, Arofat Akhundjanova, Furkan Akkurt, Gabrielė Aleksandravičiūtė, Ika Alfina, Avner Algom, Khalid Alnajjar, Chiara Alzetta, Erik Andersen, Matthew Andrews, and 633 others. 2024. Universal dependencies 2.15. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr. 2015. Conditional random fields as recurrent neural networks. In *International Conference on Computer Vision (ICCV)*.

A Decoding Algorithms

Algorithm 1 describes the unbatched version of the decoding process with timesteps halved at each round. It relies on five subroutines:

1. `sample` samples independently at each position of the input sentence according to its input categorical distribution;

Algorithm 1: Diffusion Decoding for Sequence Labelling

Input: $x \in \mathbb{R}^{N \times D}$: Token embeddings
Output: $\widehat{y} \in \mathbb{L}^N$: Predicted tags

$t \leftarrow 1024$
$y_t \leftarrow \text{sample}(\text{Cat}(\frac{1}{L}\mathbf{1}_N))$
while $t > 1$ **do**
  $\widetilde{y} \leftarrow \text{tag-embed}(y_t)$
  $\widetilde{t} \leftarrow \text{time-embed}(t)$
  $p \leftarrow \text{denoise-marginals}(x, \widetilde{y}, \widetilde{t})$
  $t' \leftarrow \lfloor \frac{t}{2} \rfloor$
  $y_t \leftarrow \text{sample}(\text{posterior}(t', t, p))$
  $t \leftarrow t'$
$\hat{y} \leftarrow y_1$
return $\hat{y}$

2. `tag-embed` and `time-embed` embed discrete values (resp. POS labels and timesteps) to their dense representations. They are implemented as look-up tables;

3. `denoise-marginals` calls the denoiser on the input sentence x from a noisy prediction $\widetilde{y}$ and compute marginal distributions of all labels at each positions. For unstructured diffusion or Mean-Field, we simply return the softmaxed logits return by the neural network implementing the denoiser. For structured diffusion we use the forward/backward algorithm.

4. `posterior` computes the posterior distribution from marginals p between timesteps t and t'.

The algorithms starts by sampling labels from the uniform distribution. Then at each relevant timestep, it performs the following operations. First the previously predicted labels, as well as the current timestep t, are embedded. Then the denoiser is called to predict a new sequence of marginal label distributions. Finally, we use the posterior distribution to sample labels at timestep $t' = \lfloor \frac{t}{2} \rfloor$. These steps are repeated recursively for timestep t', until we reach timestep 1.

B Additional Results

In Table 2, we report the results obtained using the GloVe pretrained embeddings (Pennington et al., 2014), which strengthen the validity of our approach, even when using more advanced pretrained features. In Tables 3 and 4, we compare the best baseline setups for the unigram, mean-field and CRF taggers, to the L and XL versions of our models (see F) respectively using either a transformer as an encoder, or a pre-trained multilingual transformer model, RoBerta-Large (Liu et al., 2019). Note that due to training time constraints, we only trained Diffusion-CRF, our structured diffusion model which doesn't use the mean-field approximation using RoBerta-Large as an encoder. Model sizes are described in F, while a global view of the scaling, or lack thereof for each model is presented in C. The 2 tables show that both structured and unstructured diffusion work well, however, we acknowledge that the *bitter lesson* applies here, where with an the increased parameter count of the XL model, as well as with the enriched word embeddings with RoBerta-Large, the unstructured model manages to catch up to Diff-MF.

Models	No GloVe		GloVe	
	Dev	Test	Dev	Test
`Unigram`	91.36	91.48	94.70	94.69
`MF`	93.17	93.51	95.25	95.32
`CRF`	93.84	94.02	95.39	95.48
`Diffusion-Uni`	94.47	94.74	95.71	95.75
`Diffusion-MF`	**94.72**	**94.93**	**95.79**	**95.88**

Table 2: Comparison of the scores for all models using a transformer encoder, with or without GloVe embeddings in English using UD 2.15 (EN-EWT).

	English		German		French		Dutch		AVG	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Transformer										
`Unigram`	91.36	91.48	92.85	92.38	95.12	94.58	92.03	91.94	92.84	92.60
`Meanfield`	93.17	93.51	94.12	93.88	96.49	96.02	93.92	93.84	94.42	94.31
`CRF`	93.84	94.02	94.31	94.11	96.88	96.60	94.17	94.00	94.80	94.68
`Diffusion-Uni`	94.47	94.74	**94.83**	**94.51**	**97.41**	**97.07**	**94.75**	94.48	**95.37**	95.20
`Diffusion-MF`	**94.72**	**94.93**	94.82	**94.51**	97.22	96.86	94.74	**94.54**	**95.37**	**95.21**
RoBerta-Large										
`Unigram`	98.33	98.41	97.42	96.91	98.53	98.26	97.55	97.56	97.96	97.78
`Meanfield`	98.20	98.28	97.20	96.80	98.45	98.14	97.33	97.35	97.79	97.64
`CRF`	98.32	98.41	97.33	96.85	98.54	98.26	97.47	97.48	97.92	97.75
`Diffusion-Uni`	98.42	**98.51**	97.44	**96.90**	98.55	98.27	**97.64**	97.59	98.01	97.82
`Diffusion-CRF`	98.44	98.38	97.38	96.89	98.57	**98.33**	**97.64**	97.50	98.01	97.77
`Diffusion-MF`	**98.47**	98.50	**97.48**	96.89	**98.58**	98.30	**97.64**	**97.62**	**98.04**	**97.83**

Table 3: Results for UD 2.15, average of 8 experiments using the best configurations for all 3 baseline models, and the L configuration for diffusion models).

C Ablation Studies

We find that the diffusion approach allows models to scale better with more parameters, surpassing

	English		German		French		Dutch		AVG	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Transformer										
Unigram	91.36	91.48	92.85	92.38	95.12	94.58	92.03	91.94	92.84	92.60
Meanfield	93.17	93.51	94.12	93.88	96.49	96.02	93.92	93.84	94.42	94.31
CRF	93.84	94.02	94.31	94.11	96.88	96.60	94.17	94.00	94.80	94.68
Diffusion-Uni	94.83	95.01	95.11	94.73	97.56	97.33	95.18	94.73	95.67	95.45
Diffusion-MF	**94.97**	**95.06**	**95.17**	**94.85**	**97.62**	**97.39**	**95.26**	**94.93**	**95.76**	**95.56**
RoBerta-Large										
Unigram	98.33	98.41	97.42	96.91	98.53	98.26	97.55	97.56	97.96	97.78
Meanfield	98.20	98.28	97.20	96.80	98.45	98.14	97.33	97.35	97.79	97.64
CRF	98.32	98.41	97.33	96.85	98.54	98.26	97.47	97.48	97.92	97.75
Diffusion-Uni	**98.49**	**98.52**	**97.48**	**96.97**	98.55	98.27	**97.73**	**97.65**	**98.06**	**97.85**
Diffusion-CRF	98.40	98.38	97.34	96.87	98.54	98.28	97.47	97.42	97.94	97.74
Diffusion-MF	**98.49**	98.44	97.47	96.88	**98.58**	**98.29**	97.60	97.58	98.04	97.80

Table 4: Results for UD 2.15, average of 8 experiments using the best configurations for all 3 baseline models, and the XL configuration for diffusion models).

the 3 baseline models when using a low parameter setup, but also getting increasingly better results as the parameter count increases, while the baseline models show no such improvements passed a very low ceiling after which performance actually decreases. Given that the baseline models consist principally of an encoder and a scoring MLP, we only tested their scaling when using a transformer as an encoder, whereas with the diffusion models, the encoder's size remains the same, but the parameters allotted to the diffusion and denoising processes vary, this is further explained in F. The principal limit of our scaling, to our knowledge is the limited memory in the GPUs we used.

Models	Model size				
	XS	S	M	L	XL
Transformer					
Unigram	91.17	91.36	91.16	90.87	N/A
MF	93.17	92.96	92.79	92.63	N/A
CRF	93.84	93.58	93.55	93.47	N/A
Diffusion-Uni	N/A	94.04	94.33	94.47	94.83
Diffusion-MF	N/A	**94.45**	**94.67**	**94.72**	**94.97**
RoBerta-Large					
Unigram	98.33		N/A		
MF	98.20		N/A		
CRF	98.22		N/A		
Diffusion-Uni	N/A	98.37	98.40	98.42	**98.49**
Diffusion-CRF	N/A	N/A	N/A	98.44	98.40
Diffusion-MF	N/A	**98.44**	**98.48**	**98.47**	**98.49**

Table 5: Dev scores with varying model sizes in English using UD 2.15 (EN-EWT).

Another test we did was to use different mean-field iterations, which led to increasing though diminishing gains until 10 iterations for the baseline mean-field model, and 15 iterations for Diff-MF. The effect of different numbers of iterations on speed is shown in E.

D Diffusion Tuning

N layers	1	2	3	4	6	8	10	12	14
Diff-Uni	93.76	94.45	94.62	94.72	94.80	94.84	94.86	94.90	x
Diff-MF	94.33	94.78	94.88	94.88	**94.97**	94.94	94.93	94.93	x

Table 6: Comparison of dev scores for the base and structured diffusion models with varying diffusion layer counts on UD2.15 (EN-EWT) using a transformer encoder and the EXTRA-LARGE model size.

We tested different numbers of layers for the diffusion transformer and settled on 8, which yields high results in all tested datasets while having a memory footprint small enough to ensure that both the training and evaluation can be carried out with no errors. As can be seen in 6, the unstructured model keeps getting better results with added layers, and more parameters.

E Timing Experiments

In this section we notice in that a high number of mean-field iterations as well as the diffusion itself both have a non negligible effect on speed in Table 7, however it is worth noting that the slowest training speeds we obtained for Diff-MF were about equivalent with the speed of the CRF model. Where our structured approach does slow down is in the evaluation, due to our decoding needing to perform the denoising process multiple times. Diff-CRF in particular is prohibitively slow, further validating our approximation approach with mean-fields. Similarly, our diffusion models do slow down considerably the more layers we use, as shown in 8.

F Hyperparameters

We categorize our models into 5 different size configurations, based on the parameter count they have for the main baseline, XS, S, M, L, XL, corresponding to roughly 20, 40, 60, 80 and 650 million parameters respectively. 20 million parameters are allotted to the transformer encoder, for the baseline models, unigram, mean-field, CRF, the encoder is given more parameters to reach the bigger configurations, as the encoder is all these models have, save for a scoring MLP. For the diffusion models,

MF iters	0	1	2	3	4	7	10	15
Training								
MF	2832	4277	4251	4150	4147	4008	3810	3674
Diff-MF	350	333	333	327	327	309	300	278
Evaluation								
MF	7925	9950	9917	8870	8322	9232	8679	8280
Diff-MF	209	193	191	187	182	178	166	158

Table 7: Training and evaluation speeds with different numbers of mean-field iterations on MF and Diff-MF for UD 2.15 (EN-EWT).

N_layers	1	2	3	4	6	8	10	12
Training								
Diff-Uni	677	605	522	463	350	287	240	215
Diff-MF	657	568	487	430	336	277	237	205
Evaluation								
Diff-Uni	419	344	278	249	187	159	133	120
Diff-MF	361	287	247	221	177	147	127	110

Table 8: Training and evaluation speeds for the base and structured diffusion models with varying diffusion layer counts for UD 2.15 (EN-EWT)..

the encoder does not change in size, but we change the size of the diffusion transformer used, thus in the M configuration for example, the non diffusion models have 60 million parameters allocated to their encoder, while the diffusion models have 20 million for the encoder, and 40 million for the diffusion transformer. We thus cannot have a XS diffusion baseline, as it would have 0 parameters for the diffusion, and we also decided not to test out the XL baselines for the non diffusion models, as none of them showed potential to perform any better than in the smaller baselines.

Models	Model size				
	XS	S	M	L	XL
Transformer					
Unigram	1129	999	750	640	N/A
MF	738	664	552	486	N/A
CRF	361	345	313	283	N/A
Diffusion-Uni	N/A	449	455	399	287
Diffusion-MF	N/A	343	344	322	242
RoBerta-Large					
Unigram	210		N/A		
MF	191		N/A		
CRF	152		N/A		
Diffusion-Uni	N/A	156	155	155	129
Diffusion-CRF	N/A	X	X	71	66
Diffusion-MF	N/A	140	140	141	119

Table 9: Training speeds with varying model sizes in English using UD 2.15 (EN-EWT).

Models	Model size				
	XS	S	M	L	XL
Transformer					
Unigram	4258	3798	3178	2711	X
MF	3254	3030	2740	2469	X
CRF	574	559	550	530	X
Diffusion-Uni	X	167	167	152	175
Diffusion-MF	X	122	131	127	136
RoBerta-Large					
Unigram	754		N/A		
MF	710		N/A		
CRF	358		N/A		
Diffusion-Uni	X	142	141	142	145
Diffusion-CRF	X	X	X	38	40
Diffusion-MF	X	112	113	113	118

Table 10: Evaluation speeds with varying model sizes in English using UD 2.15 (EN-EWT).