Identifying Named Entities as they are Typed

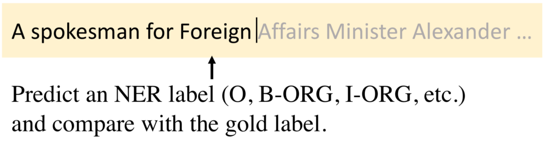
Ravneet Singh Arora **Chen-Tse Tsai** **Daniel Preoţiuc-Pietro**
Bloomberg
{rarora62,ctsai54,dpreotiucpie}@bloomberg.net

Abstract

Identifying named entities in written text is an essential component of the text processing pipeline used in applications such as text editors to gain a better understanding of the semantics of the text. However, the typical experimental setup for evaluating Named Entity Recognition (NER) systems is not directly applicable to systems that process text in real time as the text is being typed. Evaluation is performed on a sentence level assuming the end-user is willing to wait until the entire sentence is typed for entities to be identified and further linked to identifiers or co-referenced. We introduce a novel experimental setup for NER systems for applications where decisions about named entity boundaries need to be performed in an online fashion. We study how state-of-the-art methods perform under this setup in multiple languages and propose adaptations to these models to suit this new experimental setup. Experimental results show that the best systems that are evaluated on each token after its typed, reach performance within 1–5 F_1 points of systems that are evaluated at the end of the sentence. These show that entity recognition can be performed in this setup and open up the development of other NLP tools in a similar setup.

1 Introduction

Automatically identifying named entities such as organizations, people and locations is a key component in processing written text as it aids with understanding the semantics of the text. Named entity recognition is used as a pre-processing step to subsequent tasks such as linking named entities to concepts in a knowledge graph, identifying the salience of an entity to the text, identifying coreferential mentions, computing sentiment towards an entity, in question answering or for extracting relations.



A spokesman for Foreign|Affairs Minister Alexander ...

Predict an NER label (O, B-ORG, I-ORG, etc.)
and compare with the gold label.

Figure 1: An example of the proposed task and evaluation setup. After the word 'Foreign' is typed, the model immediately predicts an NER label for this word, only using left context ('A spokesman for') and the word itself. The prediction is then compared against the gold label to compute token-level F_1 score. This token's prediction will not be changed, even if the model's internal prediction for it can be revised later as more tokens are typed.

Identifying named entities as they are typed benefits any system that processes text on the fly. Examples of such applications include: a) News editors – where named entities can be highlighted, suggested (auto-completion), co-referenced or linked as the editor is typing; b) auto-correct – where named entities that are just typed are less likely to need correction as they may come from a different language or be out-of-vocabulary (OOV); c) simultaneous machine translation – where translation of OOV named entities requires different approaches; d) live speech-to-text (e.g., TV shows) – where named entities are more likely to be OOV, hence the transcription should focus more on the phonetic transcription rather than on n-gram language modelling.

This paper introduces a novel experimental setup of Named Entity Recognition systems illustrated in Figure 1. In this setup, inference about the span and type of named entities is performed for each token, immediately after it was typed. The sentence level tag sequence is composed through appending all individual token predictions as they were made. The current named entity recognition systems that are trained and evaluated to predict full sentences are likely to under-perform in this experimental setup as they: expect that right context is available,

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 976–988
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

are faced with unseen types of inputs in the form of truncated sentences and can not reconcile the final sentence-level tag sequence across the entire sentence as the result may not be a valid sequence.

The goal of this study is to present a comprehensive analysis of the task of NER in the *as-you-type* scenario, with the following contributions:

a) A novel experimental setup for conducting named entity recognition experiments, denoted as the *as-you-type* scenario;
b) Experiments with state-of-the-art sentence-level approaches to named entity recognition in the *as-you-type* setup across three languages, which indicate a 1–5 F_1 points decrease compared to sentence-level inference;
c) Tailored methods for as-you-type entity recognition models which reduce the gap to entire sentence-level inference by 9–23% compared to regular approaches;
d) An extensive analysis of existing data sets in the context of this task and model error analysis, which highlight future modelling opportunities.

2 Related Work

Named Entity Recognition is most commonly treated as a sequence labelling problem, where a prediction of whether a token is an entity and its type is done jointly for all tokens in the sentence. Over the past recent years, the dominant approach is based on recurrent neural networks, such as LSTMs (Hochreiter and Schmidhuber, 1997). These architectures use a stacked bi-directional LSTM units to transform the word-level features into distributions over named entity tags (Huang et al., 2015). Usually, an additional Conditional Random Field (CRF) (Lafferty et al., 2001) is used on the BiLSTM output in order to take into better model neighbouring tags. The tokens inputs are represented using one or a concatenation of pre-trained static word embeddings such as GloVe (Ma and Hovy, 2016), contextual word embeddings (Peters et al., 2018; Akbik et al., 2018; Devlin et al., 2019), pooled contextual word embeddings (Akbik et al., 2019b) or character embeddings trained using BiLSTMs (Lample et al., 2016) or CNNs (Ma and Hovy, 2016; Chiu and Nichols, 2016).

In addition to research on improving the performance of the NER model, other experimental setups have been proposed for this task. These include domain adaptation, where a model trained on data from a source domain is used to tag data from a different target domain (Guo et al., 2009; Greenberg et al., 2018; Wang et al., 2020), temporal drift, where a model is tested on data from future time intervals (Derczynski et al., 2016; Rijhwani and Preotiuc-Pietro, 2020), cross-lingual modelling where models trained in one language are adapted to other languages (Tsai et al., 2016; Ni et al., 2017; Xie et al., 2018), identifying nested entities (Alex et al., 2007; Lu and Roth, 2015) or high-precision NER models (Arora et al., 2019).

However, all these experimental setups assume that training is done over full length sentences. Perhaps the most related experimental setup to the one we propose for the task of entity recognition is the task of simultaneous machine translation. In this setup, the task is to generate an automatic translation in a target language as the text is being processed in the source language. The goal of the task is to produce a translation that is as accurate as possible while limiting the delay as compared to the input. Initial approaches involved identifying translatable segments and translating these independently (Fügen et al., 2007; Bangalore et al., 2012; Fujita et al., 2013) or by learning where to segment in order to optimize the system's performance (Oda et al., 2014). More recent approaches involve learning training an agent, usually using reinforcement learning, that makes a set of decisions of whether to should wait for another word from the input or write a token to the output (Gu et al., 2017). Other operations are shown to help, including predicting the verb (Grissom II et al., 2014) or the next word (Alinejad et al., 2018), better decoding with partial information (Cho and Esipova, 2016), and connecting the machine translation system to the agent's decisions (Gu et al., 2017).

Our experimental setup is different as we do not want to wait for another input token before we make a prediction about the named entity. We analyze the impact a delay has, albeit our experimental setup does not aim to combine quality and delay. The challenges are related, as the input may contain important cues for the translation or named entity decision after the current token or towards the end of the sentence, such as the verb in verb-final (SOV) languages such as German (Grissom II et al., 2014). The proposed as-you-type NER model can be useful to improve simultaneous machine translation.

3 Experimental Setup

We propose a new experimental setup for the standard task of Named Entity Recognition that would best suit real-time applications that need to process text in an online fashion.

In the regular NER experimental setup, a model is presented with a sequence of inputs $X = \{x_1, x_2, ..., x_n\}$ and it outputs a sequence of labels $Y = \{y_1, y_2, ..., y_n\}$ where $y_i \in K = \{O\} + E \times T$, where E are the set of entity types and T is the entity tag representation. Throughout the rest of the paper, we use the BIO tagging scheme ($T = \{\mathrm{B}, \mathrm{I}\}$), as this is arguably the most popular and differences in results between this tagging scheme and others, such as the BILOU scheme, are very small in practice (Ratinov and Roth, 2009). The types of entities we consider are $E = \{\mathrm{ORG}, \mathrm{PER}, \mathrm{LOC}, \mathrm{MISC}\}$.

The *as-you-type* named entity recognition setup assumes that the editor writing the text $X = \{x_1, x_2, ..., x_n\}$ needs each label prediction y_i right after the corresponding token x_i was typed. In this case, the information available for predicting y_i is only the sub-sequence $X_{1,i} = \{x_1, x_2, ..., x_i\}$. The sequence $Y = \{y_1, y_2, ..., y_n\}$ is obtained by concatenating the individual y_i predictions made for each token. Token-level micro F_1 score is used as the metric in our experiments. The evaluation process is illustrated in Figure 1.

This setup presents the model with several challenges. First, the model has no access to right context when making the prediction for each tag. However, this information is available in training. Secondly, the output sequence may contain invalid sequences of tags. For example, in the output sequence, B-ORG could be followed by I-LOC if the model decided to revise its predictions based on new information, but the evaluation setup prevents the model from revising the previous wrongly predicted tag (i.e. B-ORG). Lastly, sequences and sentences of the same length are likely to be qualitatively different and the model might need to adapt in training in order to account for these differences.

We note that this experimental setup can further be extended to account for delays in prediction, to trade-off between delays and quality or to predict entities before they are typed.

4 Data

We test our methods on three different data sets covering three different languages. We use the data sets released as part of CoNLL shared tasks in 2002 for Spanish (Tjong Kim Sang, 2002)[1] and in 2003 for English and German (Tjong Kim Sang and De Meulder, 2003).[2] The data sets contain four types of named entities: persons, locations, organizations and names of miscellaneous entities that do not belong to the previous three types. We use the standard train, dev and test splits defined for these data sets.

We chose these data sets as they are arguably the most popular data sets for performing named entity recognition and are regularly used as benchmarks for this task. We use the data sets in different languages in order to compare the impact of the language on the experimental results, identify if the commonalities and peculiarities for performing *as-you-type* entity recognition in different languages and draw more robust conclusions regarding our task setup.

4.1 Data Analysis

We perform a quantitative analysis of the data sets in order to develop some intuitions about the data in the context of the *as-you-type* experimental setup.

Sentence Length and Tag Distribution First, we study the distribution of sentence lengths. Figure 2 shows that for English, most sentences are very short (under 10 tokens) and the most frequent sentence length is two. These are expected to pose problems in the as-you-type scenario, as the context is limited. German sentences are slightly longer, while the Spanish sentences are longest, except for a spike of sentences of length one.

Figure 3 shows the distribution of entity types in sentences of varying lengths for English. For clarity, we remove MISC tags from this plot as these are infrequent. We observe there are major differences in tag distributions, especially shorter sentences (<5 tokens) containing both more entity tags as well as different tag distributions. For example, almost 30% of locations (B-LOC or I-LOC) are in sentences of length two, which are the most frequent in the English data, while in longer sentences, these are around 5%. Organizations are most frequent in sentences of length between 4 and 7 tokens, while persons are most frequent in sentences longer than 7 tokens.

Token Position and Tag Distribution To further investigate the positional bias of different tags, Fig-

[1] https://www.clips.uantwerpen.be/conll2002/ner/

[2] https://www.clips.uantwerpen.be/conll2003/ner/

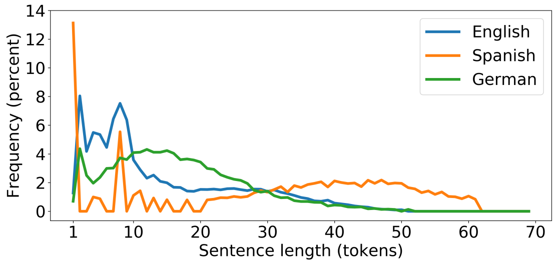


Figure 2: Distribution of sentence lengths.

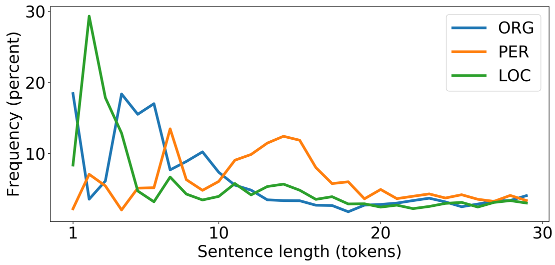


Figure 3: Distribution of entity types in terms of sentence length in the English CoNLL data set.

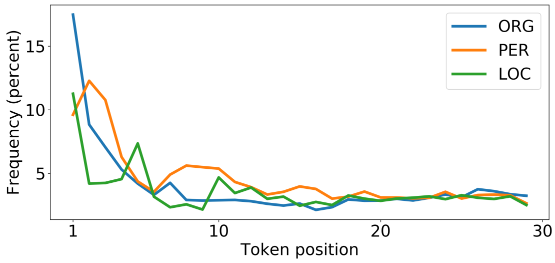


Figure 4: Distribution of entity types at each token position in the English CoNLL data set. B and I tags are merged for the same entity types and removed MISC tags as infrequent for clarity. O tag frequency can be inferred from the rest.

ure 4 shows the distribution of tags in the k-th token of the sentence. We observe that the first tokens of a sentence are much more likely to contain entities. The first position is most likely to be an ORG, with PER being the most frequent in the second to fourth positions, followed by LOC being the most prevalent for the next position, with PER again most frequent in further positions. These observations are likely to complicate the as-you-type inference for named entities, as a higher proportion of tokens will have to be inferred with no or little right context. Comparing Figures 3 and 4 shows that the model will be faced with different tag distributions when inferring the tag for the k-th token in a truncated sentence then to what it has observed in sentences of length k, which provides an intuition for our modelling strategies.

5 Methods

This section describes the methods used to perform named entity recognition in the as-you-type scenario. We use a base neural architecture that achieves state-of-the-art performance on the standard sentence-level NER task. We study its performance and observe the impact of different variants of the architecture in the *as-you-type* scenario. Following, we propose changes to the model to adapt to the as-you-type setup. We use the Flair package to conduct our experiments (Akbik et al., 2019a).[3] Implementation details and hyperparameter choices for all models are listed in the Appendix.

5.1 Base Architecture

We adopt the BiLSTM-CRF model proposed in (Huang et al., 2015) with the addition of character representation (Lample et al., 2016; Ma and Hovy, 2016). In this architecture, the word representations are fed into a bi-directional LSTM, and then the concatenated forward and backward vectors are passed through one layer of feed-forward neural network to produce a $|K|$ dimensional output for each word, where each value represents a score associated with each label. Finally, a Conditional Random Field (CRF) layer is applied to make a global decision for the entire sentence. This has the role of reconciling the independent predictions and modeling the constraints in the output space (e.g. I-PER can not follow B-ORG).

5.2 Architecture Variants

We start with studying different variants of the base neural architecture for the as-you-type scenario. The key challenge in the as-you-type setting is that the model is not presented with the future or right context (words after the current word) at test time. A natural idea is to remove information from this context during training as well. The variants we consider are based on changing the following three modeling components

Embeddings We first study the impact of different ways in which input tokens are represented. Pre-trained word embeddings obtained state-of-the-art performance on the NER task when they were introduced (Lample et al., 2016). These representations are used to initialize the word embeddings, are then fine-tuned on the training data and are concatenated

[3] https://github.com/zalandoresearch/flair

with a character-level representation of the word obtained using BiLSTMs initialized with random character embeddings.

Contextual word embeddings extend this concept to obtain different word representations for the same token in based on its context. In the standard sentence-level evaluation, contextual word embeddings were shown to obtain 2–3 F_1 points improvement on the English CoNLL data set (Peters et al., 2018; Akbik et al., 2018; Devlin et al., 2019). Without right context, the quality of word representations could be more crucial than in the standard setting. In this study, we test the performance of using classic embeddings – GloVe embeddings for English (Pennington et al., 2014) and FastText embeddings (Bojanowski et al., 2017) for German and Spanish as well as the character based contextual Flair embeddings, which achieve state-of-the-art performance on the English and German CoNLL data sets (Akbik et al., 2018). We also experimented with contextual ELMO embeddings (Peters et al., 2018) which showed slightly lower performance when compared to the Flair embeddings and hence only Flair numbers are reported due to space limitations.

However, contextual embeddings are trained with right context available. We experiment with removing this dependency from the trained embeddings and observe if this improves the performance in the as-you-type setting, as the test scenario is more similar to the training one. We note that right context is never observed in inference beyond the current token such that there is no leakage of information.

BiLSTM Bidirectional LSTM stacks two recurrent neural networks: one starts from the beginning of the sentence, and another starts from the end of the sentence. This performs better than the unidirectional variant on sentence-level experiments and shows that both types of context (left and right) are important for identifying and typing entity mentions. In the as-you-type setting, we compare unidirectional LSTM modelling left context with the bidirectional LSTM model that models both types of contexts in training.

Conditional Random Field The CRF assigns labels for words in a sentence jointly, ensuring label assignments are coherent. When running inference in the as-you-type setting, the model often sees truncated sentences which, as shown in Section 4 may have different label distributions. This discrepancy between training and test sequences may degrade the usefulness of the CRF. We experiment to see if and how the CRF is useful in the as-you-type scenario.

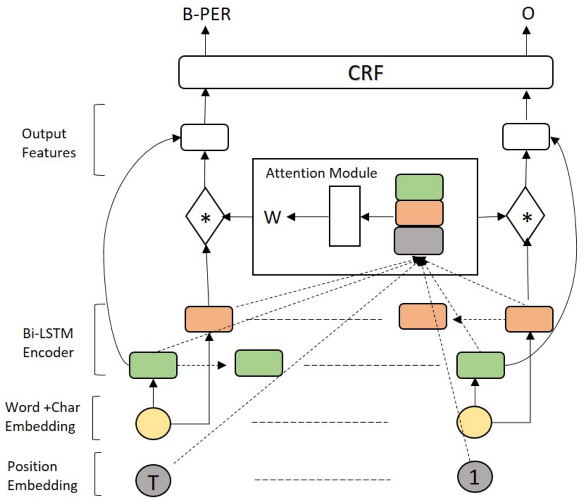


Figure 5: Architecture diagram highlighting the proposed feature weighting method described in Section 5.3.3

5.3 Model Adaptations for the As-you-type Setup

The as-you-type experimental setup presents the model with a new type of evaluation, which does not correspond to the one used in training. We thus propose the following approaches to bridge the gap between the setup and how the model is trained.

5.3.1 Weighted Loss

The model only observes the partial backward context for the tokens in the sequence during the as-you-type inference. In training, since the model has access to the entire sequence, it is likely that the model becomes too dependent on the presence and reliability of the backward features, especially for predicting the initial tokens.

In order to bias the model to be more robust to the absence or unreliable backward context, we design a new loss function that combines the original BiLSTM-CRF loss with the loss from the unidirectional LSTM features. From the latter loss, we also remove the influence of CRF as it also captures signal from the right context and, for contextual embeddings, remove the backward embeddings from the input to the LSTM. The resulting loss is:

$$L_{\text{weighted}} = L_{\text{BiLSTM-CRF}} + w * L_{\text{LSTM-NoCRF}}$$

where w is a weight treated as a hyper-parameter in our experiments.

5.3.2 Final Token Representation

The final token in a training sequence is treated in a special way in the base model. First, a stop tag is regularly used to capture the information associated with a the last token in the sequence. Secondly, the backward features for the final token in a sequence are not observed, as there is no right context, so these are initialized with a random vector. While both are useful in the regular setup as it captures a consistent pattern the final words follow, this does not hold for the as-you-type setup, where each token in a sequence will be the final token for its prediction, as partial sequences are observed.

We thus assume these choices add noise in our setup, thus we both remove the stop tag together with any transition scores to it during training and evaluation and remove the backward feature of the last token and initialize it with a zero vector.

5.3.3 Feature Weighting

The previous approach relied on the intuition of learning the trade-off between forward and backward features in order to better adapt to the inference setup. However, this trade-off is likely impacted by the position of the token in the sequence.

Thus, we explore a technique similar to (Moon et al., 2018) that allows the model to learn the importance of backward features based on the position of tokens. The is illustrated in Figure 5. We implement this using position based attention weights. We apply these weights before combining the forward and backward features (instead of concatenating) from the LSTMs as follows:

$$h_t = h_t^f + a_t * h_t^b$$

where t is the position of the token, h_t^f and h_t^b are forward and backward LSTM features at t, h_t is the new output feature at t and a_t is the attention weight for the backward features at position t. The attention weights a_t are computed as follows:

$$u_t = [h_t^f; h_t^b; p_t]; a_t = \sigma(W.u_t + b)$$

At every position t, the feature vector is calculated by concatenating the forward, backward and the positional embedding for that token. Attention weight a_t is calculated by applying attention weight matrix W followed by the sigma activation. We do not apply a_t for forward features since they are always complete and reliable even for partial sentences.

We follow the structure of positional embeddings introduced in (Vaswani et al., 2017) and defined as:

$$p_t^{2i} = sin(\frac{t}{10000^{2i/d}}); p_t^{2i+1} = cos(\frac{t}{10000^{2i/d}})$$

where d is the size of positional embedding, i is the dimension and t is the position. The values in a positional embedding are sin and cosine functions whose period is $10000^{2i/d} * 2\pi$. Positional sinusoidal embedding allows to encode longer sequences that are not present in training.

Since the right hand side context decreases as we move from left to right of a sequence in training, we would like our attention weights to consider how far a token lies from the final token in a sequence. To achieve this, we calculate position index of tokens from the end of the sentence which makes sure that a token lying at the final position always receives an index of 0, producing the same positional embedding and the input to attention weights does not fluctuate from one sequence to another.

5.3.4 Embedding Weighting

We perform a similar operation using attention at the embedding stage to trade-off between backward and forward contextual token representations. The input embeddings are calculated as follows:

$$x_t = e_t^w + e_t^f + a_t^e * e_t^b$$

$$v_t = [e_t^w; e_t^f; e_t^b; p_t]; a_t^e = \sigma(W_e.v_t + b_e)$$

where e^w are the classical word embeddings and e^f, e^b are Flair forward and backward embeddings. An architecture diagram is presented in the Appendix.

6 Results

We present the experimental results of the various NER models in the as-you-type setup, contrasting them with the regular sentence-level setup. All models are trained using the standard splits for the CoNLL data sets. The evaluation metric is token-level micro F_1, as this is reflective of our evaluation setup where each token is evaluated separately.

The top section of Table 1 shows the results of the different variants of the base architecture in Section 5.2. The overall performance drop in the as-you-type setup compared to the sentence-level setup ranges from 4.80 F1 for English to only 1.53 F1 for Spanish when comparing the best performing models. This is expected, as the as-you-type scenario is more challenging, especially for English where our data analysis from Section 4.1 showed that tokens are more prevalent in short sentences which are overall more frequent. For Spanish, where the performance difference is smallest, is where we have on average the longest sentences and in which left context alone is in most cases enough to make the correct inference.

			English		German		Spanish	
Embedding	**LSTM**	**CRF**	As-you-type	Sentence	As-you-type	Sentence	As-you-type	Sentence
Classic	⇆	✓	83.27	90.97	72.20	78.67	67.56	80.40
	→	✓	85.12	88.99	70.54	77.21	64.84	80.28
	⇆	✗	79.06	86.87	73.49	77.89	74.61	80.74
	→	✗	83.75	83.27	75.73	75.73	77.30	77.30
Flair (⇆)	⇆	✓	84.15	**92.75**	79.79	84.32	81.80	89.43
	→	✓	84.82	91.63	79.98	84.11	82.23	88.82
	⇆	✗	84.50	92.23	79.84	83.73	85.37	88.80
	→	✗	85.87	90.73	80.50	82.74	85.32	89.06
Flair (→)	⇆	✓	85.60	92.19	79.21	82.94	81.61	88.76
	→	✓	86.92	90.36	78.34	81.99	82.60	88.21
	⇆	✗	85.13	91.76	79.30	81.88	86.79	88.83
	→	✗	87.95	87.95	80.79	80.79	**87.90**	87.90
Adaptations for the as-you-type setup								
Flair (⇆)	⇆ Weighted Loss		87.77	92.46	80.39	83.71	84.13	89.48
	+ Final Token Rep		88.00	92.40	80.59	83.71	87.23	89.48
	+ Feature weighting		88.21	92.24	80.61	84.16	87.38	89.23
	+ Embedding weighting		**88.40**	92.29	**81.62**	**84.77**	87.72	**89.79**

Table 1: Evaluation results of LSTM-based NER models in the as-you-type and sentence-level evaluation setups as measured using token-level micro F_1. Arrows indicate if uni- (→) or bi-directional (⇆) training is used. Models with the best results across their setup are in **bold**. Best results within the class of methods are underlined. For classic word embeddings, we use GloVe for English, and FastText for German and Spanish. Results are averaged across three runs.

We note that in all three data sets, the best results in the as-you-type setting are obtained by matching the training setup to that of testing by only keeping a uni-directional LSTM that processes the text left to right and Flair embeddings only trained using left context. Flair embeddings trained only using left context are in all cases better than the bi-directional ones, which is natural as those embeddings would conflate information that is not available in inference. Uni-directional LSTMs perform overall better than bi-directional LSTMs by an average of a few percentage points as bi-directional LSTMs are likely learning information that will not be available in testing.

Adding the CRF hurts performance in all except one case when holding everything else constant, sometimes by wide margins (e.g. 5.3 F1 drop on Spanish with Flair forward embeddings and uni-directional LSTM). We attribute this to the mismatch between the structure of the sequences in the training data containing only full sentences, when compared to truncated sentences which can be observed by comparing Figures 3 to Figures 4.

The bottom section of Table 1 shows the results of our as-you-type adaptation strategies. All proposed methods are added on top of each other in order to study their individual contributions. For brevity, we only present the results of using Flair forward and backward embeddings as these performed best.

The changes to the last token representation and weighted loss improves on the regular bi-directional LSTM model by a substantial margin, adding between 0.45 for Spanish up to 4.25 F1 on English on the as-you type setup performance. We also notice that the sentence-level evaluation is near to the regular model performance (-0.39 for German to +0.05 for Spanish), showing that the weighted loss is able to achieve a good compromise between the representations.

Adding the feature weighting based on position marginally improves performance, between 0.02 on German to 0.21 on English. However, the weighting through attention is more effective at the embedding level improving on the previous model by between 0.19 F1 on English to 1.01 F1 on German. Overall, our as-you-type adaptation methods improve on the best variation of the base architecture on English (+0.45 F1) and German (+0.83 F1). The model is competitive on Spanish (-0.12 F1) to the Flair forward unidirectional LSTM with no CRF, albeit this is likely driven by the very long average sentence length in the Spanish data set (see Figure 2). Overall, the improvements represent between 9.3% - 23% error reduction when comparing to the best as-you-type and sentence level setups.

We highlight that an additional benefit of the proposed adaptation methods is that the model retains high performance on the sentence-level setup, in contrast with the Flair forward uni-directional LSTM, which performs between 1.89 (Spanish) and 4.34 (English) worse on the sentence-level.

Finally, the results of our proposed model are actually marginally better than the state-of-the-art approach of BiLSTM+CRF using Flair embeddings on German (+0.45 F1) and Spanish (+0.73 F1), albeit this was not the original goal of our additions. This highlights that the proposed modelling ideas are more generally beneficial as they push the model to learn more robust representations.

7 Error Analysis

Finally, we perform error analysis to identify the limitations of our approaches.

7.1 Confusion Matrix

We first study prediction difference between as-you-type and sentence-level setup. Figure 6 shows the confusion matrix on the English data set. Both models are BiLSTM-CRF with Flair embeddings, but the as-you-type model is trained with the best setting from Table 1. We can see that most confusions are between LOC and ORG: 7.9% of I-ORG tokens in the full-sentence setting are classified as I-LOC in the as-you-type setting, and 7.6% of B-ORG tokens are classified as B-LOC. Without right context, it is very challenging to distinguish these two types. For example, the data set contains many sport teams that contain location name tokens. Another noticeable difference is that the as-you-type model makes more O predictions. For instance, 5.8% of B-MISC are classified as O. This can be due to the limited availability of cues for identifying entities when right context is missing.

7.2 Positional Prediction

We expect that the size of the context impacts the quality of the inference more acutely in the as-you-type scenario when compared to the sentence-level setup. Figure 7 plots the predictive performance of three English models across each token's position. This confirms our intuition that the as-you-type setup especially impacts prediction on the first tokens in a sentence, which are more entity rich as shown in Section 4.1. However, we see that there is still a difference compared to the standard NER setup (blue curve) across all positions, confirming that indeed the right context can add more information regardless of the position of the token. The plot also highlights the performance gains of our adaptation methods at the first tokens when compared with the BiLSTM-CRF model evaluated in the as-you-type setting (orange curve).

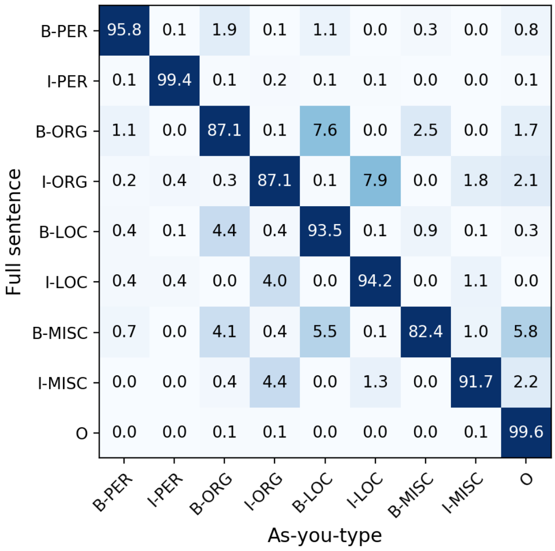


Figure 6: Confusion matrix of tag type prediction when comparing the best as-you-type and sentence-level models.

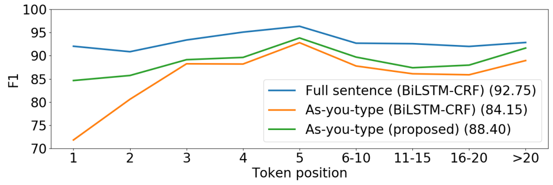


Figure 7: F_1 scores at various token positions averaged across all the sentences in the English data set. The overall performance of each model is listed in the legend.

7.3 Delayed Entity Prediction

Based on the previous results and examples, we want to study the impact of delaying the entity decision by one token. This would account for cases where the first entity word is ambiguous and also reduce the number of invalid sequences that can be produced by the as-you-type inference. For example, in the case of the 'Zimbabwe Open' entity, if the model predicted the tag of the token 'Zimbabwe' as B-LOC and after seeing the next token ('Open'), it revises this token's prediction as I-MISC, it is unable to change the B-LOC tagging, thus creating an invalid sequence, but still obtains partial credit on the token F_1 score. Delaying the decision of the first token could have allowed the model to correct its decision for the first token ('Zimbabwe') to B-MISC, resulting in a valid and more accurate sequence.

We study the possibility of delaying the prediction of a single token (not multiple tokens) by using

	English		German		Spanish	
Thr.	Tok%	F_1	Tok%	F_1	Tok%	F_1
0	0%	88.4	0%	81.62	0%	87.72
0.6	0.77%	89.18	0.97%	82.65	0.66%	88.20
0.7	1.28%	89.56	1.65%	83.09	1.15%	88.47
0.8	2.17%	90.05	2.43%	83.74	1.80%	88.66
0.9	3.25%	90.53	3.61%	84.01	2.80%	88.64
1.0	100%	91.20	100%	84.22	100%	88.91
Full sentence		92.75		84.32		89.43

Table 2: Results of the proposed wait-one policy. If the probability of prediction is less than or equal to the given threshold (Thr. column), we wait for one more token and predict again. The column Tok% indicates percentage of tokens which have a delayed prediction. The best performing model in the as-you-type setup is used. The performance of the best full-sentence model is listed in the last row for comparison purposes.

a threshold on the tag output. Table 2 shows the results of several threshold values and their impact on the total F_1.

We observe that if we delay prediction by one token for all tokens (Thr $= 1.0$), the performance is very close to the best full-sentence model, obtaining an error reduction rate of 64% (1.55 compared to 4.35) for English. Moreover, we can obtain a 50% reduction rate by only delaying the decision on 3.25% of the tokens if the downstream application deems this acceptable. These results highlight the importance of the immediate right context.

8 Untyped Entity Detection

Named entity recognition combines two different components: entity detection – identifying the named entity spans – and typing – identifying entity types. We study the impact of typing in the as-you-type setup by removing the typing information from the output labels ($E = \{\text{ENT}\}$), thus reducing the output space to $K = \{\text{B-ENT, I-ENT, O}\}$.

Results using the best as-you-type models with and without as-you-type adaptations are shown in Table 3. Comparing with the numbers in Table 1, the untyped F_1 score of as-you-type setting is much closer to the standard sentence-level evaluation, being within 1 point of F_1 for all languages. This highlights that the challenging part of the as-you-type setting is entity typing. For example, 'Zimbabwe' is a location on its own, but 'Zimbabwe Open' is an event (MISC entity type) while 'West' is usually indicative of a first location token (e.g. 'West Pacific'), but can also refer to an organization (e.g. 'West Indies' when referencing the cricket team). The proposed technique results are in this case less conclusive, which is somewhat expected as the differences in entity frequencies between full sentences and truncated sentences are smaller.

	Sentence	As-you-type	
		Original	Embedding weighting
English	97.49	97.11	97.09
German	91.37	89.29	89.49
Spanish	97.70	97.05	96.96

Table 3: Entity identification performance. The four entity types are collapsed into one type when computing token-level F_1 scores. The model for "Embedding weighting" is BiLSTM-CRF with bidirectional Flair embeddings for all three languages. For the "Original" setting, we use forward LSTM with forward Flair embeddings.

9 Conclusions

This paper introduced and motivated the as-you-type experimental setup for the popular NER task. We presented results across three different languages, which show the extent to which sentence-level state-of-the-art models degrade in this setup. Through insights gained from data analysis, we proposed modelling improvements to further reduce the gap to the regular sentence-level performance. Our error analysis highlights the cases that pose challenges to the as-you-type scenario and uncovers insights into way to further improve the modelling of this task.

This setup is tailored for end-applications such as text editors, speech-to-text, machine translation, auto-completion, or auto-correct. For text editors, the editor would be able to receive suggestions for entities inline, right after they type the entity, which can further be coupled with a linking algorithm. This would increase the user experience and efficiency of the editor, as they can make selections about entities inline (similar to a phone's autocorrect), rather than having to go back over the entire sentence after it was completed.

Another avenue of future work would be to couple the NER as-you-type with ASR data and using methods that adapt NER to noisy ASR input (Benton and Dredze, 2015) for building an end-to-end live speech to entities system.

Acknowledgements

We would like to thank Leslie Barrett, Mayank Kulkarni, Amanda Stent, Umut Topkara and the other members of the Bloomberg AI group. They provided invaluable feedback on the experiments and the paper.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019a. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019b. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 724–728, Minneapolis, Minnesota. Association for Computational Linguistics.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Beatrice Alex, Barry Haddow, and Claire Grover. 2007. Recognising nested named entities in biomedical text. In *Biological, translational, and clinical language processing*, pages 65–72, Prague, Czech Republic. Association for Computational Linguistics.

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3022–3027, Brussels, Belgium. Association for Computational Linguistics.

Ravneet Arora, Chen-Tse Tsai, Ketevan Tsereteli, Prabhanjan Kambadur, and Yi Yang. 2019. A semi-Markov structured support vector machine model for high-precision named entity recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5862–5866, Florence, Italy. Association for Computational Linguistics.

Srinivas Bangalore, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 437–445, Montréal, Canada. Association for Computational Linguistics.

Adrian Benton and Mark Dredze. 2015. Entity linking for spoken language. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 225–230, Denver, Colorado. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Jason P.C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*.

Leon Derczynski, Kalina Bontcheva, and Ian Roberts. 2016. Broad Twitter corpus: A diverse named entity recognition resource. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1169–1179, Osaka, Japan. The COLING 2016 Organizing Committee.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Christian Fügen, Alex Waibel, and Muntsin Kolss. 2007. Simultaneous translation of lectures and speeches. *Machine Translation*, 21(4):209–252.

Tomoki Fujita, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2013. Simple, lexicalized choice of translation timing for simultaneous speech translation. In *INTERSPEECH*, pages 3487–3491.

Nathan Greenberg, Trapit Bansal, Patrick Verga, and Andrew McCallum. 2018. Marginal likelihood training of BiLSTM-CRF for biomedical named entity recognition from disjoint label sets. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2824–2829, Brussels, Belgium. Association for Computational Linguistics.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1342–1352, Doha, Qatar. Association for Computational Linguistics.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. 2017. Learning to translate in real-time with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume*

1, Long Papers, pages 1053–1062, Valencia, Spain. Association for Computational Linguistics.

Honglei Guo, Huijia Zhu, Zhili Guo, Xiaoxun Zhang, Xian Wu, and Zhong Su. 2009. Domain adaptation with latent semantic association for named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 281–289, Boulder, Colorado. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

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*, pages 282–289.

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.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867, Lisbon, Portugal. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Seungwhan Moon, Leonardo Neves, and Vitor Carvalho. 2018. Multimodal named entity disambiguation for noisy social media posts. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2000–2008, Melbourne, Australia. Association for Computational Linguistics.

Jian Ni, Georgiana Dinu, and Radu Florian. 2017. Weakly supervised cross-lingual named entity recognition via effective annotation and representation projection. *arXiv preprint arXiv:1707.02483*.

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Optimizing segmentation strategies for simultaneous speech translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 551–556, Baltimore, Maryland. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado. Association for Computational Linguistics.

Shruti Rijhwani and Daniel Preotiuc-Pietro. 2020. Temporally-informed analysis of named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7605–7617, Online. Association for Computational Linguistics.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Chen-Tse Tsai, Stephen Mayhew, and Dan Roth. 2016. Cross-lingual named entity recognition via wikification. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 219–228, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Jing Wang, Mayank Kulkarni, and Daniel Preotiuc-Pietro. 2020. Multi-domain named entity recognition with genre-aware and agnostic inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8476–8488, Online. Association for Computational Linguistics.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 369–379, Brussels, Belgium. Association for Computational Linguistics.

A Appendices

A.1 Implementation and Hyperparameters

To train our models we use Stochastic Gradient Descent with a learning rate of 0.1 and mini batch size of 32. The LSTM model includes 1 layer of LSTM with hidden state size 256. We also employ a dropout of 0.5 for the LSTM layer. For the positional embeddings, the dimension d is set as 100 for feature weighting and 1000 for embedding weighting. We tried different dimensions between 100 and 2000. w for weighted loss is identified as 1.5 for English and 1 for German and Spanish from the dev set. For W we considered values between 0 and 2. All the models are trained for 70 epochs and the best model is selected based on the token-level F-1 score[4] on dev set. We perform manual hyper-parameter selection and the final performance is reported based on the 5-10 runs of the best hyper-parameter setting.We use Flair's standard settings for English.

All the models are trained on nvidia GPU and overall training for 70 epochs takes around 5-6 hours. This run-time complexity is very close to the complexity achieved by the the Flair implementation for standard NER training.

Numbers reported in (Akbik et al., 2018) are generated by training models on combined train and dev sets, hence they are higher than the numbers we report when training only on the training data. We also report token-level F1, rather than entity-level F1, which leads to results that are not directly comparable with (Akbik et al., 2018).

A.2 Visualization of Attention Weights

To better understand the impact of positional attention weights, we visualize and compare the feature-level attention weights for different tokens on a few hand-picked English sentences. Figure 8 highlights tokens using different color intensities. Higher intensity represents a larger weight value and hence a stronger influence of backward context. First, it is evident that tokens in the first position rely more heavily on backward features in the absence of any forward context, which is further reflected in the higher attention weights achieved by these tokens. Moreover, first tokens of multi-token entities such as Persons ('Nabil Abu Rdainah'), Organizations ('NY Rangers') and Events ('Zimbabwe Open') are assigned larger weights due to a high influence of immediate next tokens. Also, quite often the last token in the sentences are weighted lower which can be attributed to the positional information captured by the attention weights.

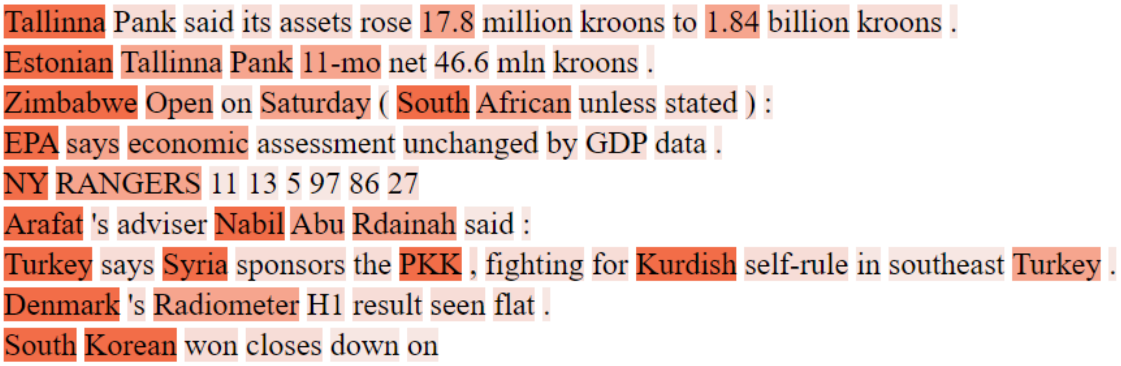
A.3 Performance on Dev Set

To facilitate reproducibility of results, Table 4 reports the development set performance of the base model (Bi-LSTM CRF Flair) and the proposed model for both as-you-type and sentence level setups.

A.4 Parameters

Table 5 lists different trainable parameters used in the model along with their sizes.

[4] https://sklearn-crfsuite.readthedocs.io/en/latest/tutorial.html#evaluation



Tallinna Pank said its assets rose 17.8 million kroons to 1.84 billion kroons .
Estonian Tallinna Pank 11-mo net 46.6 mln kroons .
Zimbabwe Open on Saturday (South African unless stated) :
EPA says economic assessment unchanged by GDP data .
NY RANGERS 11 13 5 97 86 27
Arafat 's adviser Nabil Abu Rdainah said :
Turkey says Syria sponsors the PKK , fighting for Kurdish self-rule in southeast Turkey .
Denmark 's Radiometer H1 result seen flat .
South Korean won closes down on

Figure 8: Sample attention weights from the English CoNLL Data Set.

	English		**German**		**Spanish**	
Model	As-you-type	Sentence	As-you-type	Sentence	As-you-type	Sentence
Base Model	88.16	96.08	82.74	86.48	81.34	87.54
Proposed Model	92.54	96.43	83.71	86.69	85.75	87.78

Table 4: Performance on Conll Dev set for both Bi-LSTM-CRF Flair and the proposed final model

Parameter	Size
GloVe Embedding Dimension (English)	100
Fast-text Embedding Dimension (Spanish, German)	300
Flair Embedding Size	2,000
Feature-Level Positional Embedding Size	100
Embedding-Level Positional Embedding Size	1,000
LSTM output size	100
Bi-Directional LSTM	1,680,800
Linear Output Layer	200 * 9
CRF Transition Matrix	6 * 6
Feature Level Attention Matrix (W)	300*1
Embedding Level Attention Matrix (W_e) (English)	5,100 * 1 + 1
Embedding Level Attention Matrix (W_e) (Spanish, German)	5,300 * 1 + 1

Table 5: Number of parameters for different components of our models. When not explicitly mentioned, parameters are for models in all three languages.