Pushing the Limits of Low-Resource Morphological Inflection

Antonios Anastasopoulos and Graham Neubig
Language Technologies Institute, Carnegie Mellon University
{aanastas,gneubig}@cs.cmu.edu

Abstract

Recent years have seen exceptional strides in the task of automatic morphological inflection generation. However, for a long tail of languages the necessary resources are hard to come by, and state-of-the-art neural methods that work well under higher resource settings perform poorly in the face of a paucity of data. In response, we propose a battery of improvements that greatly improve performance under such low-resource conditions. First, we present a novel two-step attention architecture for the inflection decoder. In addition, we investigate the effects of cross-lingual transfer from single and multiple languages, as well as monolingual data hallucination. The macro-averaged accuracy of our models outperforms the state-of-the-art by 15 percentage points.[1] Also, we identify the crucial factors for success with cross-lingual transfer for morphological inflection: typological similarity and a common representation across languages.

1 Introduction

The majority of the world's languages are categorized as synthetic, meaning that they have rich morphology, be it fusional, agglutinative, polysynthetic, or a mixture thereof. As Natural Language Processing (NLP) keeps expanding its frontiers to encompass more and more languages, modeling of the grammatical functions that guide language generation is of utmost importance.

In the case of morphologically-rich languages, explicit modeling of the inflection processes has significant potential to alleviate issues created by data scarcity and the resulting lack of vocabulary coverage. Especially on low-resource, under-represented languages and dialects, the potential for *impact* is much higher. For example, speech
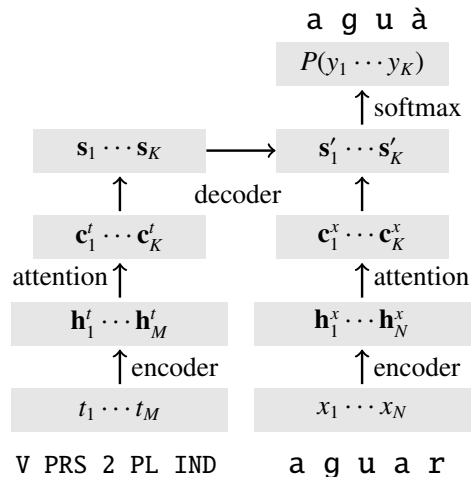

Figure 1: Visualization of our proposed two-step attention architecture. The decoder first attends over the tag sequence $\mathbf{T}$ and then uses the updated decoder state $\mathbf{s}'$ to attend over the character sequence $\mathbf{X}$ in order to produce the inflected form $\mathbf{Y}$. (Example from Asturian.)

recognition (Foley et al., 2018) and predictive keyboards (Breiner et al., 2019) for under-represented languages, if they exist, largely still rely on unigram lexicons with performance inferior to the sophisticated language models used in high-resource ones. Good inflection models would be invaluable for predictive text technology in morphologically-rich languages as they could effectively enable proper handling of the huge vocabulary.

Additionally, they could be very useful for building educational applications for languages of under-represented communities (along with their inverse, morphological analyzers). Encouraging examples are the Yupik morphological analyzer (Schwartz et al., 2019) and the Inuktitut educational tools from the respective Native Peoples communities.[2] The social impact of such applications can be enormous, effectively raising the status of the languages slightly closer to the level of

---

[1]Our code is available at `https://github.com/antonisa/inflection`.

[2]`http://www.inuktitutcomputing.ca/`

the dominant regional language.

Morphological inflection has been thoroughly studied in monolingual high resource settings, especially through the recent SIGMORPHON challenges (Cotterell et al., 2016, 2017, 2018). Low-resource settings, in contrast, are relatively under-explored. One promising direction and the main focus of the SIGMORPHON 2019 challenge (McCarthy et al., 2019b) is cross-lingual training, which has been successfully applied in other low-resource tasks such as Machine Translation (MT) or parsing.

In this work we focus in this cross-lingual setting for low-resource morphological inflection and propose several simple, yet effective approaches to mitigating problems caused by extreme lack of data which, put together, *improve accuracy by 15 percentage points over a state-of-the-art baseline*. This is achieved through the combination of a novel decoder architecture, a training regime that alleviates the need for costly structural biases that force attention monotonicity, and a data hallucination technique. We also present thorough ablations and identify the crucial factors for success with our approach.

Our system was the best performing system in the 2019 SIGMORPHON shared task on morphological inflection when evaluated on accuracy, while it ranked third when evaluated with average Levenshtein distance.

2   Task Definition and Approach

Morphological inflection is the process that creates grammatical forms (typically guided by sentence structure) of a lexeme/lemma. As a computational task it is framed as mapping from the lemma and a set of morphological tags to the desired form, which simplifies the task by removing the necessity to infer the form from context. For an example from Asturian, given the lemma `aguar` and tags `V;PRS;2;PL;IND`, the task is to create the indicative voice, present tense, 2nd person plural form `aguà`.

Let $\mathbf{X} = x_1 \dots x_N$ be a character sequence of the lemma, $\mathbf{T} = t_1 \dots t_M$ a set of morphological tags, and $\mathbf{Y} = y_1 \dots y_K$ be an inflection target character sequence. The goal is to model $P(\mathbf{Y} \mid \mathbf{X}, \mathbf{T})$.

Our approach consists of three major components. First, we propose a novel two-step attention decoder architecture (§2.1). Second, we augment the low-resource datasets with a data halluci-

nation technique (§2.2). Third, we devise a training schedule (§2.3) that substitutes structural biases for attention monotonicity.

2.1   Model Architecture

Our models are based on a sequence-to-sequence model with attention (Bahdanau et al., 2015). In broad terms, the model is composed of three parts: an encoder, the attention, and a recurrent decoder. In the setting of the inflection task, there is an additional input provided (the set of morphological tags) which requires an additional encoder.

A visualization of our model is shown in Figure 1. First, an encoder transforms the input character sequence $x_1 \dots x_N$ into a sequence of intermediate representations $\mathbf{h}_1^x \dots \mathbf{h}_N^x$. A second encoder transforms the set of morphological tags $t_1 \dots t_M$ into another sequence of states $\mathbf{h}_1^t \dots \mathbf{h}_M^t$:

$$\mathbf{h}_n^x = \text{enc}^x(\mathbf{h}_{n-1}^x, x_n) \quad \text{and} \quad \mathbf{h}_m^t = \text{enc}^t(\mathbf{T}).$$

In our implementation, we use a single layer bidirectional recurrent encoder for the lemma, and a self-attention encoder (Vaswani et al., 2017) for encoding the tags as there is no inherent order (e.g. left-to-right) in their presentation. In preliminary experiments, a self-attention lemma encoder proved hard to train, while a recurrent tag encoder yielded quite competitive results.

Next, we have attention mechanisms that transform the two sequences of input states into two sequences of *context vectors* via two matrices of *attention weights* (k is the current decoder time step):

$$\mathbf{c}_k^x = \left[\sum_n \alpha_{kn}^x \mathbf{h}_n^x \right] \qquad \mathbf{c}_k^t = \left[\sum_m \alpha_{km}^t \mathbf{h}_m^t \right].$$

Finally, the recurrent decoder computes a sequence of *output states* in a two-step process, from which a probability distribution over output characters can be computed:

$$\mathbf{s}_k = \mathbf{s}_{k-1}' + \mathbf{c}_k^t$$
$$\mathbf{s}_k' = \text{dec}(\mathbf{s}_{k-1}', \mathbf{c}_k^x, y_{k-1})$$
$$P(y_k) = \text{softmax}(\mathbf{s}_k').$$

The attention mechanisms produce their weights as in Luong et al. (2015), with $\mathbf{v}^x$, $\mathbf{v}^t$, $\mathbf{W}_{\alpha^x}^{s'}$, $\mathbf{W}_{\alpha^t}^s$, $\mathbf{W}_{\alpha^x}^h$, and $\mathbf{W}_{\alpha^t}^h$ being parameters to be learned:

$$\alpha_{km}^t = \text{softmax}(\mathbf{v}^t \tanh([\mathbf{W}_{\alpha^t}^s \mathbf{s}_{k-1}'; \mathbf{W}_{\alpha^t}^h \mathbf{h}_m^t]))$$
$$\alpha_{kn}^x = \text{softmax}(\mathbf{v}^x \tanh([\mathbf{W}_{\alpha^x}^{s'} \mathbf{s}_k; \mathbf{W}_{\alpha^x}^h \mathbf{h}_n^x])).$$

The two-step attention process essentially first uses the decoder's previous state $\mathbf{s}_{k-1}'$ as the query for attending over the tags. Then, it creates a tag-

informed state $\mathbf{s}'_k$ by adding the tag context $\mathbf{c}^t_k$ to the previous state. The tag-informed state is then used as the query for attending over the source characters and produce the context $\mathbf{c}^x_k$. The last step is then to update the recurrent state and produce the output character. Ultimately, we desire that the provided tag set guides the generation, which also means influencing the attention over the characters of the lemma.

Additional Structural Biases for Attention Incorporating structural biases in the model's architecture or in the training objective can lead to improvements in performance, especially for tasks where the attention mechanism is expected to behave similarly to an alignment model, like MT. This idea has been successfully applied to the inflection task and is at the core of the state-of-the-art model (Wu and Cotterell, 2019).

One bias we deem important is *coverage* of all input characters and tags from the attentions. Intuitively, this entails encouraging the model to "look at" the whole input. We take the approach of Cohn et al. (2016) and add two regularizers over the final attention matrices, encouraging them to also sum to one column-wise:

$$-\lambda \parallel \Sigma_j a^t_{jm} - \mathbb{I} \parallel_2 \qquad -\lambda \parallel \Sigma_j a^x_{jn} - \mathbb{I} \parallel_2$$

Another bias that we incorporate encourages the Markov assumption over attention/alignments. Briefly, this means that if the i-th source character/word is aligned to the j-th target one ($i \leftarrow j$), then alignments $i{+}1 \leftarrow j{+}1$ or $i \leftarrow j{+}1$ are also quite likely. In a neural architecture this can be approximated by providing the attention weight vector from the previous timestep as input to the function that computes the attention weights. We refer the reader to Cohn et al. (2016) for exact details.

Adversarial Language Discriminator When training multilingual systems, encouraging the encoder to learn language-invariant representations can often lead to improvements (Xie et al., 2017; Chen et al., 2018), as it forces the model to truly work in a multilingual setting. We achieve that by introducing a language discriminator (Ganin et al., 2016). This additional component receives the last output of the (bi-directional) intermediate lemma representations $\mathbf{h}^x_N$ and outputs a prediction y_l of the source language such that $y_l{=}\text{softmax}(\text{MLP}(\mathbf{h}^x_N))$.

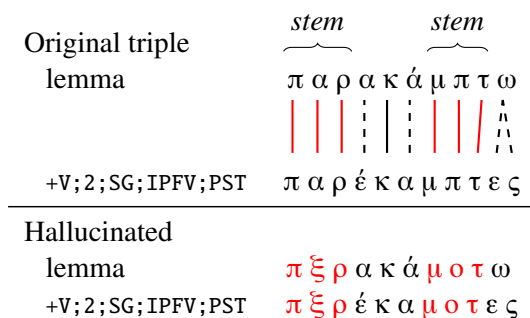The discriminator is trained to predict the language by minimizing a standard cross-entropy

Figure 2: Example of our hallucination process (Greek). The lemma and inflected forms are aligned at the character level. The inside of *stem*-considered parts (highlighted) are substituted with random characters, creating hallucinated triples (bottom).

loss $\mathcal{L}_l$ similar to Lample et al. (2018). However, in order to encourage the encoder to learn language-invariant representations, we reverse the gradients flowing from that component into the encoder during back-propagation.

2.2 Data Hallucination

Low-resource language datasets are usually too small to allow for proper learning with neural networks. A major issue in our case is *label bias*. Put simply, the character decoder will overly prefer outputting common character sequences. However, with just 50 examples to learn from, the output character n-gram distribution will hardly match the real one, because the majority of the n-grams will have zero probability. In order to mitigate this issue, we augment our training sets with hallucinated data.

In most languages morphological reinflection is realized by adding, deleting, or modifying prefixes or suffixes over a stem that is mainly unchanged. Though we do not have prior information regarding the stem or affixes of our data we can use character alignment to approximately compute them. We use the alignment method from the SIGMOR-PHON 2016 baseline (Cotterell et al., 2016) to align the lemmata and the inflected forms. For each example, we consider as part of the stem *any* sequence of three or more[3] consecutive lemma characters that are aligned to the exact same characters in the inflected form.

Now, for each such region considered as a "stem", we randomly substitute its inside (not start or end) characters with other characters from the language's alphabet. Note that we do not change

---

[3]This heuristic number is largely arbitrary and could potentially be tuned to different values for each language.

the length of the region, though allowing for such variation could possibly lead to further improvement. The substitution characters are sampled uniformly for the alphabet, rather than attempting to sample from a more informed distribution, which has potential for further improvements. Overall, we hallucinated 10,000 examples for each low-resource language, creating an additional hallucinated dataset $\mathcal{H}$.

A visualized example of our hallucination process is outlined in Figure 2. Out of the three regions with matching aligned characters (thick lines), we identify two with length equal to three or more. In the hallucinated example (bottom of the figure), we sample random characters for the inside of such regions.

Silfverberg et al. (2017) have proposed a data hallucination method conceptually quite similar to ours, which treats the *single* longest common continuous substring between lemma and form as the stem. Their approach would be effectively similar to ours for languages with affixal stem-invariant morphology, but it would likely fail in more complicated morphological phenomena like apophony, stem alternation (conceptually similar to infix morphology) or in the root-and-pattern morphology of semitic languages. Consider the apophony example from the past participle form `gschwommen` of the lemma `schwimmen` in Swiss German: our approach would treat both the `schw` and the `mmen` regions as a stem, as opposed to only considering one.[4] Note that neither of the two approaches are suitable for phenomena such as suppletion (e.g. the inflection of the Spanish verb `ir` 'to go' into `fue` 'went') or phenomena like the reduplication pattern of Indonesian noun plurals as in `kuda` 'horse', `kuda-kuda` 'horses' (Sneddon et al., 2012).

2.3 Training Schedule

Our training schedule attempts to balance two desires: helping the model to (1) learn to copy, and (2) learn cross-lingually with a particular focus on the low-resource language. To achieve this, we split training into three phases: warm-up, cross-lingual training, and low-resource fine-tuning.

Phase 1: Warm-Up As several previous works have noted, learning to copy is crucial. Unlike other proposed models, though, our model does not include any structural biases that encourage

---

[4]Most likely, the desired parts are `schw` and `mm`.

Model	Accuracy	Median
Wu and Cotterell (2019)	48.5	45.5
this work	48.8	67.0
+$\mathcal{H}$	60.1	66.6
+$\mathcal{H}$ + $\mathcal{L}_l$	60.8	66.0
+multi-language transfer	**63.8**	64.0
oracle	68.2	74.0

Table 1: Macro-averaged accuracy over 100 language test pairs. Our best model outperforms the baseline by 15 percentage points.

copying. Instead, we rely on an additional copying task in a warm up period.

We transform each training triple $[\mathbf{X}, \mathbf{T}, \mathbf{Y}]$ into two additional triples that encourage copying: $[\mathbf{X}, \texttt{COPY}, \mathbf{X}]$ and $[\mathbf{Y}, \mathbf{T}, \mathbf{Y}]$. Using both the input and output sequences for the copying task allows the use of slightly more diverse data. Note that we only use the correct tags when copying the inflected form. For copying the lemma we use a specialized tag (`COPY`). Additional improvements could be achieved if one knew the exact tags that match the lemmata. But this could vary by language: an English verb's lemma is its infinitive and has a `V;NFIN` tag, while a verb lemma in Modern Greek is its `V;1;SG;IPFV;PRS` form.

In this stage we use a relatively large batch size (10) in order to encourage more coarse updates. In most cases, the model achieves extremely high copying accuracy after a couple of warm-up epochs, so we stop the warm up stage when copying accuracy exceeds 75%. At this point the attention mechanism over the source characters has learned to be monotonic. In contrast, the model of Wu and Cotterell (2019) requires a dynamic programming method that forces strict monotonicity, with an additional (non-parallelizable) computational cost $O(|x|^2)$ throughout training.

Phase 2: Cross-lingual training In the main training phase we use both high- and low-resource language data (including any hallucinated data). If not using hallucinated data, we up-sample the low-resource data in order to match the size of the high-resource ones. Furthermore, with probability 0.30 we also sample copying tasks to intersperse throughout the training epoch. This ensures the source-character attention keeps being monotonic.

Phase 3: Fine-tuning The last phase is inspired by fine-tuning, or continued training, as applied to

Model	Dev Accuracy (macro-averaged)
Wu and Cotterell (2019):	
0th-order soft attention	37.1
0st-order hard attention	43.5
0th-order monotonic attention	45.5
1st-order monotonic attention	51.3
two-step attention (this work)	
− warm-up, copy task	31.2
+ warm-up, copy task	48.0
+ structural biases	48.7
+ scheduled sampling	49.1
+ minibatch schedule	49.4
with hallucinated data:	
$+\mathcal{H}$	$82.6^{\dagger}$
$+\mathcal{H} + \mathcal{L}_l$	$85.4^{\dagger}$
+ensemble (three models)	$85.6^{\dagger}$

Table 2: Our proposed two-step attention architecture outperforms almost all baselines on the development set (but with higher gains on the test set). Additional biases and scheduled sampling contribute improvements. †: results with hallucinated data are not directly comparable, as dev data were used for hallucination.

cross-lingual MT e.g. by Neubig and Hu (2018) or to domain adaptation for MT e.g. by Luong and Manning (2015). The setting is nearly identical to the second phase, except we only use the low-resource language data for training and do not use the copying task. Furthermore, we substitute teacher forcing for scheduled sampling (Bengio et al., 2015), where with probability 50% the input to the next step of the decoder is not the gold one, but its previous prediction. This technique allows the model to become more robust to its own mistakes, effectively limiting the effect of exposure bias. It is worth noting that at this point, the learning rate is typically quite small, so we reduce the batch size to a single instance. In most cases, though, the improvements on development set accuracy are marginal (1 to 2%).

2.4 Inference and Implementation Details

The performance of inflection systems is typically evaluated with exact-match token-level accuracy, as well as character-level Levenshtein distance.[5] Hence, during training we continuously evaluate

---

[5]Due to space constraints, we do not provide Levenshtein distance results. They generally inversely correlate with token-level accuracy.

the model's performance on the development set with both metrics. Consequently we store three checkpoints: the one that achieved the highest accuracy, the one that reached the lowest Levenshtein distance, and the one that improved on both metrics over previous dev evaluations. In a few cases these three checkpoints coincide, but this is rather rare. We ensemble these three models with equal weights for producing our final predictions.

All our models are implemented in DyNet (Neubig et al., 2017). Each model is trained on a single CPU, as each training run requires less than 1GB of RAM and typically concludes within 3 to 4 hours. We provide additional hyperparameter details in the Appendix.

3 Empirical Results

Our experiments are conducted on the SIGMORPHON 2019 challenge datasets (McCarthy et al., 2019a). The dataset consists of 100 pairs between (mostly related) high-resource transfer languages and 43 low-resource test languages with examples taken from the Unimorph database (Kirov et al., 2018). The low-resource languages typically have about 100 training examples, plus 50–100 (or in very few cases, 1000) examples in the development set. In contrast, most high-resource language training sets include 10,000 instances.

Test Accuracy Our main results are summarized in Table 1. Our models significantly outperform the baseline, evaluated with macro-averaged accuracy over the 100 language pairs. Our novel architecture performs slightly better than the baseline without any additional data. Importantly, including the hallucinated datasets boosts total accuracy by 10 percentage points, while transfer from multiple languages further improves to a state-of-the-art accuracy of 63.8% – higher than any system submitted to the shared task.

It is worth noting that all of these improvements are not uniform across languages/pairs: without hallucinated training data, our model's median is notably larger than its average, implying that for a few language pairs our model is under-trained and under-performs (in particular, pairs with Yiddish, Votic, and Ingrian as test languages). Nonetheless, if one had access to an oracle that would allow them to select the best performing model out of all the different settings, they would achieve an oracle accuracy of 68.2%.

L1	L2	Genetic dist.	L1+L2	$+\mathcal{L}_l$	$+\mathcal{H}$	$+\mathcal{L}_l + \mathcal{H}$	$\mathcal{H}$
latin	czech	0.86	15.0	26.0	71.4	68.0	**77.4**
bengali	greek	0.88	22.4	16.4	70.5	70.6	**71.6**
sorani	irish	1.0	20.3	18.6	**66.3**	64.6	65.6
italian	ladin	0.30	48	54	**74**	**74**	74
latvian	lithuanian	0.25	17.1	23.2	48.4	48.4	**50.5**
english	murrinhpatha	N/A	**36**	2	6	20	20
italian	neapolitan	0.10	70	70	83	83	**84**
urdu	old english	0.88	23.8	18.5	43.4	40.4	**44.3**
slovene	old saxon	0.89	10.7	14.7	**52.3**	50.5	50.5
russian	portuguese	0.93	34.5	22.2	**88.8**	88.4	87.7
swahili	quechua	1.0	14.2	12.8	**92.1**	91.6	91.6
portuguese	russian	0.93	25.6	17.5	**76.3**	74.6	74.3
kurmanji	sorani	N/A	16.2	13.6	**69.0**	64.3	66.7
zulu	swahili	1.0	46	52	**81**	80	76
kannada	telugu	0.75	76	72	**94**	90	**94**
Average		0.75	31.72	28.90	67.77	67.23	68.55

Table 3: Results with a single transfer language. Monolingual data hallucination is crucial due to the distance of the languages. In some cases cross lingual transfer should be avoided in favor of a purely monolingual setting ($\mathcal{H}$).

Architecture Ablations We focus on the architecture of the model and the structural biases we introduced, using the development set for evaluation. Table 2 presents the macro-averaged accuracy for the baseline (Wu and Cotterell, 2019) and the different versions of our model.[6]

The first thing to note is the importance of the warm-up period in our training schedule and the use of the copying task. Our model neither handles copying in any explicit way nor encourages the attention to be monotonic. Without the warm-up period and the additional copying tasks, our model's dev accuracy is worse than any of the baselines.

On the other hand, our two-step attention trained with the additional copying task already improves over most baselines without any of the additional biases, with a dev accuracy of 48. The best baseline model, with an average dev accuracy of 51.3, is better than our models, but this difference is not reflected on the test set (see Table 1) where our model performs slightly better. This discrepancy could be due to optimizer or early stopping decisions, but we leave a more thorough investigation for future work.

We attribute the success of our model to two factors, the first being our novel architecture. Compared to a single attention over the concatenated tag and lemma sequences, our two-step attention has the advantage of two distinct attention mechanisms, which capture the inherently different properties of the tag and the lemma character sequences. Another advantage is the two-step

process that guides the lemma attention with the tags. Disentangling the two attentions *and* ordering them in an intuitive way makes it easier for them to learn their respective tasks.

The second factor is, we suspect, a slightly better choice of hyperparameters: our models are smaller than the baseline ones. Although we did not tune our hyperparameters extensively, a few experiments on a couple of language pairs showed that increasing the model size hurt performance under these extremely data-scarce conditions.

Each of the additional techniques we tested further contributed a few accuracy points. The attention biases add 0.7% points and scheduled sampling in the third training phase further adds 0.4% points. The different batching schedule with large batch sizes in the beginning but smaller towards the end of training helps a little more in terms of accuracy, but most importantly it speeds up training. Table 2 also reports the development set accuracy when using hallucinated data. Although the large improvements are also proportionally reflected in the test set, these numbers are not directly comparable to the rest, as the development set data were used in the hallucination process.

4 Analysis

We analyze the results over various groupings of languages to elucidate the properties of our models over the 100 quite diverse language pairs. We use typological information from the URIEL database (Littell et al., 2017) in this analysis.

Single Language Transfer We first focus on the test languages for which a *single* transfer language

---

[6]Table 2 includes unpublished results kindly shared by Wu and Cotterell, the authors of the baseline.

L1	L2	L1+L2	$+\mathcal{L}_l$	$+\mathcal{H}$	$+\mathcal{L}_l+\mathcal{H}$	$\mathcal{H}$
turkish		81	77	80	81	
persian		55	63	74	69	
bashkir	azeri	57	59	66	67	66.7±0.9
uzbek		47	55	74	70	
all		84	71	83	**87**	
urdu		42	32	66	**67**	
sanskrit		44	38	66	65	
hindi	bengali	49	52	67	65	63.7±4.0
greek		42	46	65	**67**	
all		49	50	64	62	
turkish		87	80	85	**89**	
bashkir	crimean	59	60	70	69	
uzbek	tatar	60	60	72	67	71.3±1.1
all		82	81	88	80	
finnish		38	36	34	40	
hungarian		28	32	32	38	
estonian	ingrian	32	32	32	38	34.6±2.3
all		38	**44**	36	36	
finnish		54	50	58	56	
hungarian		42	46	58	52	
estonian	karelian	42	42	54	58	52.6±1.1
all		50	46	54	**64**	
basque		46	40	70	76	
slovak		58	62	74	76	
czech	kashubian	54	64	78	66	74.0±2.3
polish		66	78	78	**80**	
all		70	72	78	**80**	
estonian		27	27	35	34	
hungarian		28	30	35	33	
finnish	livonian	30	26	34	35	33±1
all		26	25	**36**	**36**	
italian		34	25	**48**	42	
arabic		18	26	41	38	
hebrew	maltese	20	21	47	45	45.6±5.6
all		29	28	40	46	
danish	middle	68	58	78	80	
dutch	high	70	62	74	82	
german	german	72	68	**86**	82	76.7±6.0
all		70	70	84	78	
danish		18	25	44	46	
dutch	north	28	22	**47**	46	
english	frisian	22	26	**47**	42	43±1
all		23	25	43	46	
asturian		58	49	77	74	
spanish		47	55	78	76	
french	occitan	41	52	**80**	**80**	78.3±3.5
all		52	61	78	**80**	
russian	old	40	39	39	**64**	
polish	church	41	41	59	58	
bulgarian	slavonic	38	42	44	56	57.3±1.5
all		41	42	**64**	56	
sanskrit		20	16	46	44	
persian	pashto	36	28	39	48	46.5±3.5
all		29	30	46	**49**	
bashkir		67	64	73	69	
uzbek		55	45	67	72	
turkish	tatar	81	79	82	75	73.0±4.6
all		81	79	**84**	83	
Average (over all pairs)		49.00	48.48	59.36	60.18	57.9

Table 4: Results with multiple transfer languages (sample). The best performing system per target (L2) language is **highlighted**. We repeated the hallucinated-data-only experiments as many times as potential transfer (L1) languages, hence the $\mathcal{H}$ column reports average accuracy ± standard deviation.

was provided, with detailed results presented in Table 3. Generally, the average genetic distance between the transfer and the task language is quite high (0.75) for these pairs. It is easy to observe that the larger the typological distance, the larger the improvement from adding hallucinated data. In fact, excluding the languages with no typological information available, there is strong corre-

lation (ρ=0.6) between genetic distance and improvement from hallucinated data.

For cross-lingual transfer to be useful, the languages need to be at least somewhat related and share similar characteristics. A prime example is transfer from Italian for Neapolitan, which achieves a 70% accuracy without any additional synthesized data. In the same vein, the same condition is necessary for the adversarial language discriminator to have impact, as using it on extremely distant language pairs leads to worse performance (e.g. Russian-Portuguese, Bengali-Greek, or Urdu-Old English). This is expected, as forcing language invariant representations across vastly different languages is analogous to representing a bimodal distribution with its mean.

The results on Kurmanji-Sorani (Northern Kurdish-Central Kurdish) seem to be a valid counter-example to the above statement, i.e. the two languages are related,[7] but cross-lingual transfer without hallucinated data performs poorly, achieving a mere 16.2 accuracy. The reason for this discrepancy lies in the characters: Kurmanji is written in the Latin alphabet, while Sorani uses the Arabic one.[8] The lack of any similar representations across the languages is too hard to overcome even with the adversarial language discriminator.[9]

Multiple Transfer Languages For most low-resource languages and especially dialects, there exist several possible candidate transfer languages that can be related enough to satisfy the similarity constraint. We present extensive ablations on such cases in Table 4 with results on the rest of the SIGMORPHON language pairs.[10]

We again observe positive correlations between the language genetic proximity and the performance of cross-lingual transfer, even with all the transfer and test languages being related. For example, transfer from (distant) Basque to Kashubian performs about 10 percentage points worse than transfer from (related) Slovak or Czech, which in turn perform worse than transfer from Polish (more closely related to Kashubian than the

---

[7] A reviewer pointed out that Kurmanji and Sorani might actually be more distant than the politically-motivated term "Kurdish dialects" might imply.

[8] Kurmanji is also sometimes written with the Cyrillic or the Nashk Arabic scripts, but our data are Latin-only.

[9] We did not attempt to address this issue, by e.g. increasing the loss weight, but we leave this for future work.

[10] We present a sample due to space constraints. The discussion pertains to all language pairs. The full table is available in the Appendix.

others). Transfer for North Frisian using Danish is also 10 percentage points worse than transfer from the more closely related Dutch. It is worth noting that using the hallucinated data reduces the effect of the genetic distance between languages, with the standard deviation across the results within the same test language becoming much smaller.

A very interesting case of cross-lingual transfer is Maltese, which is a semitic language and hence genetically close to Hebrew and Arabic. Surprisingly, we obtain better results when transferring from Italian. Again, a script discrepancy could be the main reason, also considering that the root and pattern morphology is only partially expressed in the scripts of Hebrew and Arabic, whereas it is fully expressed (by writing all the vowels) in Maltese. We should also point out that genetic similarity might not be enlightening enough. As Hoberman and Aronoff (2003) point out, the productive verbal morphology in Maltese has become affixal due to borrowings from Romance languages. To further exacerbate the situation, all provided train, dev, and test examples are verbs (no nouns or adjectives), providing an explanation to this seemingly counter-intuitive result.

Another interesting case is that of cross-lingual transfer for Bengali, with the potential languages varying from very related (Sanskrit, Hindi) to even only distantly related (Greek). Nevertheless, there is notably little variance in the performance of the systems. We believe that again the culprit is the difference in writing systems between all selected transfer and test language, which does not allow our system to leverage cross-lingual information: our Bengali data use the Bengali script, the Urdu dataset is in the Arabic one, Hindi and Sanskrit use Devanagari, and Greek uses the Greek alphabet.

We also present analytic results with cross-lingual transfer from all transfer languages from the suggested SIGMORPHON pairs.[11] In 14 out of the 29 test languages, our best-performing model is trained on multiple transfer languages. For instance, using Turkish, Persian, Bashkir, and Uzbek data for transfer to Azeri leads to a 6 point improvement over any single-language-transfer result. A potential explanation is that a dialect/language has indeed been influenced by multiple languages. Another reason could lie in the increased amount of data and potential regulariza-

tion effects. We suspect the truth lies in the union of those factors, but nonetheless we conclude that **whenever available, transfer from multiple related languages can further improve accuracy**.

In our experiments we used all transfer languages that the SIGMORPHON organizers proposed in the 2019 challenge. On the one hand, a more sophisticated data selection process could likely yield improvements. For example, Yiddish is primarily based in High German and it has elements from Hebrew, Aramaic, as well as Slavic languages, but we did not test how transfer from these languages might perform. Moreover, in order to remain faithful to the SIGMORPHON challenge, we used some distant languages that probably worsen the results (e.g. Greek for Bengali).

Also, alphabet divergence issues still need to be addressed. For instance, we suspect that our accuracy in Yiddish (which uses the Hebrew alphabet with Yiddish orthography) could be greatly improved by finding some type of mapping between its orthography and the Latin script that most of its related languages use. The same could hold for transfer for the central Asian languages (Tatar, Turkmen, Azeri among others) which use a variety of the Latin, Cyrillic, or Arabic scripts.

Lastly, we experimented with a completely monolingual setting, using just the low-resource and hallucinated language data (columns $\mathcal{H}$ in Tables 3 and 4). For fairer comparison to cross-lingual transfer, we repeated the hallucination process as many times as candidate transfer languages and we report the mean and standard deviation of the test set accuracy. This baseline is extremely competitive, lagging only a few points behind the $\mathcal{L} + \mathcal{H}$ combination. Encouragingly, this entails that hallucination is a viable option for entire language families without a single high-resource representative or low-resource isolates.

Interpretability We find that the attention matrices can help understand our model's predictions. A visualization of two examples is shown in Figure 3, showcasing the interpretability advantage of the disentangled two-step attentions. In the Kazakh example the tag attention clearly identifies the suffixes да (that marks plural) and рды (that marks the accusative case). The Greek example is a great one of how the two-step process allows the tags to guide the lemma attention. Due to the SBJV tag, the model does not use the lemma until the necessary particle να has been generated. Conse-

---

[11]We do not use *all* languages for transfer in a test language. For instance, when testing on Occitan 'all' stands for training on Asturian, Spanish, and French.
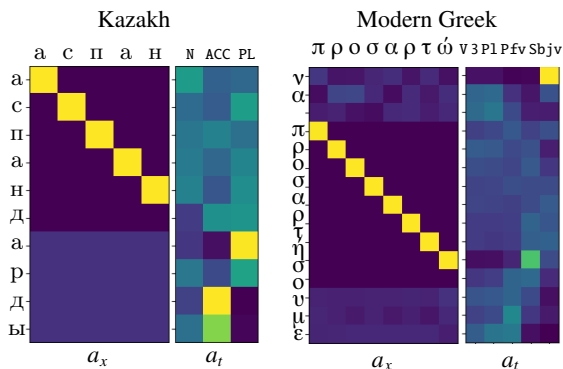
Figure 3: Attention visualization examples. The inflected form is generated from top to bottom.

quently, the lemma attention properly copies the stem, and then the tag attention attends first over PFV and then over PL and 3 in order to construct the correct suffix for perfective and 3$^{\text{rd}}$ person plural.

5 Related Work

The inflection task in high-resource settings has been extensively studied through the SIGMORPHON shared tasks. Notably, the best models explicitly model copying and hard monotonic attention (Aharoni et al., 2016; Aharoni and Goldberg, 2017) with the previous state-of-the-art forcing strict monotonicity (Wu and Cotterell, 2019). We instead achieve state-of-the-art with a cheaper approach that simply intermixes a copying task which also encourages monotonicity.

Data augmentation for inflection has been explored by Bergmanis et al. (2017) and Zhou and Neubig (2017) among others. The work of Silfverberg et al. (2017) is the most similar to ours, but as we already discussed, it has a few shortcomings that our approach addresses.

Kann et al. (2017) have identified typology as playing a role for cross-lingual transfer, but they measure language similarity using lexical overlap. We attest that this data-based measure is less informative and more suspect to variation, so we instead use the genetic typological information to quantify correlations between performance improvements and language distance.

Our novel two-step process decoder architecture bares similarities with multi-source models (Anastasopoulos and Chiang, 2018; Zoph and Knight, 2016) which provide two contexts from two encoded sources to the decoder. A similar disentangled encoding was also used by Ács (2018) for their SIGMORPHON 2018 submission. We in fact experimented with this architecture but pre-

liminary results on the development sets showed that our two-step architecture achieved better performance. Interestingly, the second-best performing system (Peters and Martins, 2019) at SIGMORPHON 2019, which also ranked first in terms of Levenshtein distance, also uses decoupled encoders to separately encode the lemma and the tags; this further cosolidates our belief that such an approach is superior to using a single encoder for the concatentated sequence of the tags and lemma. The main difference to our model is that they do not use our two-step decoder process, while they substitute all softmax operations with sparsemax (Martins and Astudillo, 2016), yielding interpretable attention matrices very similar to ours. The use of sparsemax in conjunction with our two-step decoder process, as well as along our data hallucination technique, presents a promising direction towards even better results in the future.

6 Conclusion

With this work we advance the state-of-the-art for morphological inflection on low-resource languages by 15 points, through a novel architecture, data hallucination, and a variety of training techniques. Our two-step attention decoder follows an intuitive order, also enhancing interpretability. We also suggest that complicated methods for copying and forcing monotonicity are unnecessary. We identify language genetic similarity as a major success factor for cross-lingual training, and show that using many related languages leads to even better performance. Despite this significant stride, the problem is far from solved. Language-specific or language-family-specific improvements (i.e. proper dealing with different alphabets, or using an adversarial language discriminator) could potentially further boost performance.

Acknowledgements

References

Judit Ács. 2018. BME-HAS system for CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proc. CoNLL–SIGMORPHON*, pages 121–126, Brussels. Association for Computational Linguistics.

Roee Aharoni and Yoav Goldberg. 2017. Morphological inflection generation with hard monotonic attention. In *Proc. ACL*.

Roee Aharoni, Yoav Goldberg, and Yonatan Belinkov. 2016. Improving sequence to sequence learning for morphological inflection generation: The BIU-MIT systems for the SIGMORPHON 2016 shared task for morphological reinflection. In *Proc. SIGMORPHON*.

Antonios Anastasopoulos and David Chiang. 2018. Leveraging translations for speech transcription in low-resource settings. In *Proc. INTERSPEECH*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. ICLR*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proc. NIPS*.

Toms Bergmanis, Katharina Kann, Hinrich Schütze, and Sharon Goldwater. 2017. Training data augmentation for low-resource morphological inflection. *Proc. SIGMORPHON*.

Theresa Breiner, Chieu Nguyen, Daan van Esch, and Jeremy O'Brien. 2019. Automatic keyboard layout design for low-resource Latin-script languages. arXiv:1901.06039.

Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2018. Adversarial deep averaging networks for cross-lingual sentiment classification. *Transactions of the Association for Computational Linguistics*, 6:557–570.

Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. Incorporating structural alignment biases into an attentional neural translation model. In *Proc. NAACL-HLT*.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sebastian Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proc. CoNLL–SIGMORPHON*.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017. CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proc. CoNLL SIGMORPHON*.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The SIGMORPHON 2016 shared task—morphological reinflection. In *Proc. SIGMORPHON*.

Ben Foley, Josh Arnold, Rolando Coto-Solano, Gautier Durantin, T Mark Ellison, Daan van Esch, Scott Heath, František Kratochvíl, Zara Maxwell-Smith, David Nash, et al. 2018. Building speech recognition systems for language documentation: The Co-EDL endangered language pipeline and inference system (Elpis). In *Proc. SLTU*.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *JMLR*.

Robert D Hoberman and Mark Aronoff. 2003. The verbal morphology of Maltese. *Language Acquisition and Language Disorders*, 28:61–78.

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2017. One-shot neural cross-lingual transfer for paradigm completion. In *Proc. ACL*.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sebastian J. Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: Universal Morphology. In *Proc. LREC*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *Proc. ICLR*.

Patrick Littell, David R. Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. Uriel and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proc. EACL*.

Minh-Thang Luong and Christopher D Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proc. IWSLT*.

Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proc. EMNLP*.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proc. ICML*, pages 1614–1623.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sebastian Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019a. The SIGMORPHON 2019 shared task: Crosslinguality and context in morphology. In *Proc. SIGMORPHON*.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Miikka Silfverberg, Sebastian J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019b. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, et al. 2017. DyNet: The dynamic neural network toolkit. arXiv:1701.03980.

Graham Neubig and Junjie Hu. 2018. Rapid adaptation of neural machine translation to new languages. In *Proc. EMNLP*.

Ben Peters and André F. T. Martins. 2019. IT–IST at the SIGMORPHON 2019 shared task: Sparse two-headed models for inflection. In *Proc. SIGMORPHON*, Florence, Italy.

Lane Schwartz, Emily Chen, Benjamin Hunt, and Sylvia LR Schreiner. 2019. Bootstrapping a neural morphological analyzer for St. Lawrence Island Yupik from a finite-state transducer. In *Proc. Comput-EL3*.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. *Proc. SIGMORPHON*.

James Neil Sneddon, K Alexander Adelaar, Dwi N Djenar, and Michael Ewing. 2012. *Indonesian: A comprehensive grammar*. Routledge.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. NeurIPS*.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. arXiv:1905.06319.

Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. 2017. Controllable invariance through adversarial feature learning. In *Proc. NeurIPS*, pages 585–596.

Chunting Zhou and Graham Neubig. 2017. Morphological inflection generation with multi-space variational encoder-decoders. In *Proc. SIGMORPHON*.

Barret Zoph and Kevin Knight. 2016. Multi-source neural translation. In *Proc. NAACL-HLT*.

A Hyperparameters

Here we list all the hyperparameters of our models:

- Encoder/Decoder number of layers: 1
- Character/Tag Embedding size: 32
- Recurrent State Size: 100
- LSTM Type: CoupledLSTM
- Attention Size: 100
- Target and Source Character Embeddings are Tied
- Tag Self-Attention size: 100
- Tag Self-Attention heads: 1

Here we list the optimizer settings:

- Optimizer: SimpleSGD
- Starting learning rate: 0.1
- Learning Rate Decay: 0.5
- Learning Rate Decay Patience: 6 epochs
- Maximum number of epochs: 20, 40, 40 (for each training phase)
- Minibatch Size: 10, 10, 2 (for each training phase).

L1	L2	Genetic dist.	L1+L2	$+\mathcal{L}_l$	$+\mathcal{H}$	$+\mathcal{L}_l + \mathcal{H}$	$\mathcal{H}$
latin	czech	0.86	15.0	26.0	71.4	68.0	**77.4**
bengali	greek	0.88	22.4	16.4	70.5	70.6	**71.6**
sorani	irish	1.0	20.3	18.6	**66.3**	64.6	65.6
italian	ladin	0.30	48	54	**74**	**74**	**74**
latvian	lithuanian	0.25	17.1	23.2	48.4	48.4	**50.5**
english	murrinhpatha	N/A	**36**	2	6	20	20
italian	neapolitan	0.10	70	70	83	83	**84**
urdu	old english	0.88	23.8	18.5	43.4	40.4	**44.3**
slovene	old saxon	0.89	10.7	14.7	**52.3**	50.5	50.5
russian	portuguese	0.93	34.5	22.2	**88.8**	88.4	87.7
swahili	quechua	1.0	14.2	12.8	**92.1**	91.6	91.6
portuguese	russian	0.93	25.6	17.5	**76.3**	74.6	74.3
kurmanji	sorani	N/A	16.2	13.6	**69.0**	64.3	66.7
zulu	swahili	1.0	46	52	**81**	80	76
kannada	telugu	0.75	76	72	**94**	90	**94**
Average		0.75	31.72	28.90	67.77	67.23	68.55

Table 5: Results with a single transfer language. Monolingual data hallucination is crucial due to the distance of the languages. In some cases cross lingual transfer should be avoided in favor of a purely monolingual setting ($\mathcal{H}$).

B Complete Result Tables

Table 5 lists all results for test languages with a single candidate transfer language. Results on test languages with multiple candidate languages (both single-language-transfer, and transfer with all candidate suggested languages) are listed in Tables 6 and 7.

L1	L2	L1+L2	$+\mathcal{L}_l$	$+\mathcal{H}$	$+\mathcal{L}_l+\mathcal{H}$	$\mathcal{H}$
turkish		81	77	80	81	
persian		55	63	74	69	
bashkir	azeri	57	59	66	67	66.7±0.9
uzbek		47	55	74	70	
all		84	71	83	**87**	
urdu		42	32	66	**67**	
sanskrit		44	38	66	65	
hindi	bengali	49	52	67	65	63.7±4.0
greek		42	46	65	**67**	
all		49	50	64	62	
welsh		54	55	83	86	
irish		48	52	80	**88**	
albanian	breton	53	59	78	81	78.2±2.5
all		62	55	81	82	
hebrew	classical	86	85	95	95	
arabic	syriac	83	82	92	95	94±0
all		87	85	95	**96**	
irish		18	24	24	20	
welsh	cornish	26	**28**	**28**	24	24±0
all		22	16	22	24	
turkish		87	80	85	**89**	
bashkir	crimean	59	60	70	69	71.3±1.1
uzbek	tatar	60	60	72	67	
all		82	81	88	80	
spanish		55	55	81	**83**	
italian	friulian	56	55	78	77	79.0±2.6
all		64	58	78	**83**	
finnish		38	36	34	40	
hungarian		28	32	32	38	
estonian	ingrian	32	32	32	38	34.6±2.3
all		38	**44**	36	36	
adyghe		92	92	93	**96**	
armenian	kabardian	78	77	86	80	90.6±4.0
all		95	91	93	90	
finnish		54	50	58	56	
hungarian	karelian	42	46	58	52	52.6±1.1
estonian		42	42	54	58	
all		50	46	54	**64**	
basque		46	40	70	76	
slovak		58	62	74	76	
czech	kashubian	54	64	78	66	74.0±2.3
polish		66	78	78	**80**	
all		70	72	78	**80**	
turkish		**86**	80	74	66	
bashkir	kazakh	68	84	74	70	73.3±3.0
uzbek		66	60	78	70	
all		76	78	80	78	
bashkir		82	78	80	84	
uzbek	khakas	84	80	72	76	72.6±6.4
turkish		90	90	80	82	
all		84	**92**	86	84	
czech		5.4	5.0	20.6	42.0	
romanian	latin	7.9	9.0	18.8	41.3	41.9±1.4
all		3.6	7.7	40.1	**44.1**	
estonian		27	27	35	34	
hungarian	livonian	28	30	35	33	33±1
finnish		30	26	34	35	
all		26	25	**36**	**36**	

Table 6: Multiple transfer language results (part 1).

L1	L2	L1+L2	$+\mathcal{L}_l$	$+\mathcal{H}$	$+\mathcal{L}_l+\mathcal{H}$	$\mathcal{H}$
italian		34	25	**48**	42	
arabic	maltese	18	26	41	38	45.6±5.6
hebrew		20	21	47	45	
all		29	28	40	46	
danish	middle	68	58	78	80	
dutch	high	70	62	74	82	76.7±6.0
german	german	72	68	**86**	82	
all		70	70	84	78	
dutch	middle	36	42	32	36	
german	low	**46**	26	32	38	36.6±3.0
danish	german	30	24	30	30	
all		32	42	36	38	
danish		18	25	44	46	
dutch	north	28	22	**47**	46	43±1
english	frisian	22	26	**47**	42	
all		23	25	43	46	
asturian		58	49	77	74	
spanish	occitan	47	55	78	76	78.3±3.5
french		41	52	**80**	**80**	
all		52	61	78	**80**	
russian	old	40	39	39	**64**	
polish	church	41	41	59	58	57.3±1.5
bulgarian	slavonic	38	42	44	56	
all		41	42	**64**	56	
irish		2	4	**12**	2	
belarusian	old irish	10	10	10	6	6±2
welsh		4	6	10	6	
all		4	4	8	6	
sanskrit		20	16	46	44	
persian	pashto	36	28	39	48	46.5±3.5
all		29	30	46	**49**	
welsh		46	34	64	64	
irish	scottish	58	62	**68**	66	58±2
latvian	gaelic	58	36	58	66	
all		60	64	62	66	
bashkir		67	64	73	69	
uzbek	tatar	55	45	67	72	73.0±4.6
turkish		81	79	82	75	
all		81	79	**84**	83	
bashkir		82	76	82	88	
arabic		64	66	84	80	
turkish	turkmen	82	88	90	**92**	80±4
uzbek		66	74	86	78	
all		**92**	90	84	88	
estonian		12	17	23	27	
finnish	votic	24	20	26	28	26.3±2.1
hungarian		10	10	24	**30**	
all		17	20	28	29	
dutch		36	39	44	**49**	
english	west	36	25	45	43	47±1
danish	frisian	33	33	42	43	
all		40	38	43	45	
danish		50	54	55	56	
dutch	yiddish	49	50	56	55	55.3±0.6
german		53	54	**57**	54	
all		55	55	55	54	
Average (over all pairs)		49.00	48.48	59.36	60.18	57.9

Table 7: Multiple transfer language results (part 2).