# Unsupervised Learning of Morphology

by

**Müge Kural**

A Dissertation Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Data Science

KOÇ ÜNİVERSİTESİ

September 9, 2022

**Unsupervised Learning of Morphology**

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

**Müge Kural**

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

Prof. Deniz Yuret (Advisor)

_____

Prof. Tunga Güngör

_____

Asst. Prof. Gözde Gül Şahin

Date: _____

*Dedicated to all who fail and try, fail and try, fail and try...*

# ABSTRACT

**Unsupervised Learning of Morphology**
**Müge Kural**
**Master of Science in Data Science**
**September 9, 2022**

Unsupervised learning of morphological rules is one of the expected abilities of natural language processing (NLP) models since children learn these rules during their native language acquisition without supervision. Based on this expectation, we present a comprehensive experimental setup for evaluating the morphological learning of several unsupervised models such as Autoencoders (AE), Variational Autoencoders (VAE), Character-level Language Models (CharLM) and Vector Quantized Variational Autoencoders (VQVAE) at the following tasks: probing for morphological features, morphological segmentation and morphological reinflection. In our study, we show that for probing, all models outperform baselines with an indication of encoding morphological knowledge; for morphological segmentation, VAE and CharLMs have comparable performances to unsupervised SOTA models; for morphological reinflection, VQVAE with multiple codebooks has the ability to identify the lemma and suffixes of a word and turns out to be a good candidate to perform inflectional tasks.

# ÖZETÇE

**Yüksek Lisans Tez Başlığı**
**Müge Kural**
**Veri Bilimleri, Yüksek Lisans**
**9 Eylül 2022**

Morfolojik kuralların denetimsiz öğrenilmesi, doğal dil işleme (NLP) modellerinin beklenen yeteneklerinden biridir, çünkü insanlar bu kuralları ana dili edinimleri sırasında denetimsiz olarak öğrenmektedirler. Bu beklentiye dayanarak, Autoencoders (AE), Variational Autoencoders (VAE), Character-level Language Models (CharLM) ve Vector Quantized Variational Autoencoders (VQVAE) gibi denetimsiz birçok modelin morfolojik öğrenmesini değerlendirmek için kapsamlı bir deneysel kurulum sunuyoruz. Çalışmamızda morfolojik özelliklerin sondalanması, morfolojik segmentasyon ve morfolojik yeniden çekim deneylerine yer veriyoruz. Deneylerimizde, tüm modellerin, morfolojik bilgiyi kodladığının bir göstergesi olarak sondalama deneylerinde taban modellerden daha iyi performanslar gösterdiğini; morfolojik segmentasyon için VAE ve CharLM'lerin SOTA modelleriyle karşılaştırılabilir performanslara sahip olduğunu; birden fazla kod kitabına sahip VQVAE'nin, bir kelimenin kökünü ve son eklerini belirleme yeteneğine sahip olduğunu ve bu açıdan morjolojik çekimsel görevlerini gerçekleştirmek için iyi bir aday olduğunu gösteriyoruz.

# ACKNOWLEDGMENTS

I would first like to express my gratitude to my advisor Prof. Deniz Yuret, who always supported and guided me throughout my studies, for teaching me to ask what is important and why it is important...I also want to thank Prof. Tunga Güngör and Asst.Prof. Gözde Gül Şahin for participating in my thesis committee.

I want to thank my dear and lovely friend Sadra Safadoust for always being there for me; without his support and help, this thesis work would not have been realized.

I also want to thank my friends in KUIS AI, especially Ahmet Canberk Baykal, Hilal Güven, Batuhan Özyurt, Vahideh Hayyolalam, Kerem Özfatura and Burak Can Biner for always motivating and supporting me during the hard times. I also want to thank my KUIS AI Morphology team friends, Emre Can Açıkgöz and Tilek Chubakov, for their support and valuable feedbacks and showing me how essential teamwork is.

I will always be thankful to Fatih, who believed in me and supported me in pursuing my goals.

Finally and foremost, I want to thank my dear family, my mother, my father, and my dear sister Zehra Hande Kural, who always believed in me, for their unconditional love and support.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# ABBREVIATIONS

| | |
|---|---|
| AE | Autoencoder |
| VAE | Variational Autoencoder |
| VQVAE | Vector Quantized Variational Autoencoder |
| LSTM | Long Short Term Memory |
| CharLM | Character Level Language Model |

# Chapter 1

# INTRODUCTION

In a language, morphological rules determine how words should be constructed to convey their intended meanings. For example, if we want to express a present participle of a verb *to hike*, it is *hiking* but not *hikeing*; we should delete the last vowel. Or when we discuss whether a goal can be achieved or not, we talk about its *achievability* (achieve+able+ity) but not its *achieveityable* (achieve+ity+able); we should order the suffixes in correct way. Learning these rules occurs unsupervised during human language acquisition. As we learn a language, we are not exposed to its morphological rules explicitly. According to [Clark, 2017], during our language acquisition, we must first learn to analyze the structure of words heard, identify their stems and affixes, map consistent meanings onto them, and then begin to use those stems and affixes in new combinations. When we grasp the definition of learning as "a process that leads to change, which occurs as a result of experience and increases the potential for improved performance and future learning" [Ambrose et al., 2010], we can come up with evaluations for machine learning model learning as we do for human learning. This thesis aims to evaluate various *unsupervised* machine learning models in the context of different computational tasks requiring morphological knowledge.

A machine learning model uses examples to learn a task given by optimizing a predefined objective function and improves its ability to perform the task for the upcoming examples. While the task and the examples can be from any field in the world, such as text, vision, audio, speech, etc., our way of giving examples may also change. It is possible to provide examples with labels; that way of learning is called *supervised learning*; or we can provide examples without labels, which is

called *unsupervised learning*.

In unsupervised learning, since there are no labels for data, the tasks can be defined as what can only be solved within the data. A model can be asked to cluster the data, reduce the dimension of the data, reconstruct the data from a low-dimensional representation, corrupt then recover the data, and so on. While in both cases, we are the ones who define the objectives (and the tasks) that need to be achieved by the model, those two ways diverge in terms of what we *actually* expect from the model. In supervised learning, we generally expect the model to label the upcoming examples precisely as we do, and this can be our only motivation to train a model. For example, one can aim for a model to translate Turkish sentences into English. However, when we give only Turkish sentences without any labels and select an applicable unsupervised task, such as reconstructing the sentence from a low-dimensional representation, our primary goal is not obtaining the sentence that we already have. Instead, we are interested in how the model deciphers the data, such as what the model learns about distinguishing features forming data, which examples share the same features, or how these features can be used to generate that data.

Deciphering the observed data without direct supervision is precisely what we humans do for our language acquisition. We are born, hear sounds, see objects, and listen to the conversations in our environment; throughout the years, we learn our language. Each world language has its rules; no matter where we are born, we can learn them without direct supervision. This brings us to the question: despite the richness and diversity of languages worldwide, how can children acquire the skill of learning a language so quickly? [Goldsmith et al., 2017] If we think of unsupervised models that study text data, we can come up with an analogy of a child and an unsupervised model. This analogy excites us, like many researchers, to understand language learning in unsupervised models. Before moving to the ways of evaluating the learning of language in unsupervised models, we first narrow the area of language learning for our study.

Language researchers divide the language into five basic components: phonology (study of the sound system in a language), morphology (study of word structures in a

language), syntax (study of how to arrange words in a language), semantics (study of the meaning of words and word combinations in a language) and pragmatics (study of how to use language in social communications). This thesis focuses on the branch that studies structures of words: morphology and its learning in unsupervised models.

Morphology, which takes its name from Greek words, (*morph: shape, form* + *ology: a study of something*) is a subbranch of linguistic studies that is interested in the internal structure of words in a language. A word is formed from smaller pieces called *morpheme*. A morpheme is a minimal unit that contributes to the word's meaning. For example in the word *untouchable*, there are three morphemes: *un-*: indicating not, *touch*: the root and *-able*: indicating "an ability". The morphemes are classified into two categories: free morphemes and bound morphemes. Free morphemes, as the name implies, are morphemes that can occur independently, such as dog, hall, or touch. On the other hand, bound morphemes must be attached to another morpheme to have meaning so they appear in words. *Un-* is, for example, a bound morpheme that appears only when combined with other morphemes to create a word. Bound morphemes are usually affixes, such as prefixes and suffixes. There are two types of bound morphemes: derivational and inflectional. They differ primarily in how they relate to words. When combined with a root, a derived morpheme changes a word's semantic meaning or part of speech (PoS), such as a noun, adjective, or verb. For example in the word *beautiful*, *-ful* is a derivational morpheme since it changes the noun word beauty into adjective. Another example from Turkish, in word *sanatçı*, *-çı* is a derivational morpheme that converts word meaning from art to an artist. Inflectional morphemes, on the other hand, do not change the word's PoS class or its meaning. These morphemes change the form of a word by altering its tense, aspect, mood, person, or number in a verb or the case, number, or gender of a noun, adjective, or pronoun. For example in the word *kitapları*, *-lar* changes the number of the noun *kitap* where *-ı* changes its case to accusative. Briefly, the morphemes are the main elements of morphological study, and the rules that govern the formation of words from them are the main concerns of morphology.

With computation and morphology in mind, researchers have come up with various computational approaches to studying morphology over the years. There are two motivations behind this research: testing theories about morphology learning in humans and investigating practical applications such as spell checking and correction for everyday life, automatic speech recognition, and statistical machine translation. The studies are firstly dominated by the idea of [Koskenniemi, 1983] two-level morphology model, which implies that morphology requires more than a simple concatenation of morphemes since phonological alternations occur with the concatenation of them. As a sample, we can think of Turkish words *evimden* (from my house) and *yurdumdan* (from my dorm). Both words include morphemes that convey the meaning of movement from the possessed word. However, one morpheme sequence appears as *+im+den* and the other *+um+dan* because of the phonological rules in Turkish. Two-level morphology, divides this process in two-levels: first level, which consider the *lexical* forms of morphemes *ev+Hm+DAn* and *yurt+Hm+DAn*, and second level, which considers the phonological alternations and creates the *surface* form of word: *evimden* and *yurdumdan*. The two-level process has been studied with finite-state transducers [1] (a machine that is given a sequence of symbols as input and outputs a string by exploiting defined rules) for many world languages throughout the years. [Oflazer, 1993] also provides a morphological analyzer for Turkish based on the two-level formalism.

Nevertheless, morphologically rich languages contain numerous ambiguous cases, and two-level finite-state transducers provide all possible analyses. For example, a Turkish word *masalı* can be analyzed using the accusative and possessive forms of the stem 'masal' (tale) and the +With form of the stem 'masa' (table). In order to deal with this problem, statistics and machine learning techniques were first introduced, providing disambiguation. Afterward, the use of these techniques in the field has become widespread. Both supervised and unsupervised learning techniques have been used. Both modeling the language acquisition process and reducing the need for annotated data are important reasons to study unsupervised learning settings.

[1] For further notes: [Cahill, 2016]

To evaluate unsupervised models from a morphological perspective, we should first define what we expect from a model that "learned" morphology. As stated in [Goldsmith et al., 2017], some of the questions that an unsupervised learner of morphology should be able to answer are: "What are the component morphemes of each word? Are there alternative forms (allomorphs, e.g., *-ler, -lar* in Turkish) of any morphemes, and if so, under what conditions is each used? Are there alternative forms of morphemes that need to be explained by reference to phonological generalizations in the language? Are there inflectional paradigms in the language? If so, how many independent dimensions (or morphosyntactic features) are active in each paradigm? What combinations of morphosyntactic feature specifications are permitted in the language, and how is each such combination realized morphologically? Are there processes of derivational morphology present in the language? How productive is each of the processes discovered?"

Some of these questions are intended to be answered within academic competitions. Morpho Challenges [Kurimo et al., 2010] were held between 2005-2010, and the systems were asked to segment input words into morphemes. For the first competition, systems were asked to segment Turkish, Finnish, and English words into their morphemes. Later, German was added to the list of languages, and the systems were also asked to pair up morpheme variations. Semi-supervised learning and Arabic language learning were also permitted last year. SIGMORPHON tasks followed morphological evaluations in the literature. Since 2016, SIGMORPHON organized shared tasks on morphological reinflection [Cotterell et al., 2016a] [Cotterell et al., 2017], [Cotterell et al., 2018] Cognitively Plausible Morphological Inflection [Pimentel et al., 2021] Unsupervised Paradigm Completion [Kann et al., 2020], Unsupervised Paradigm Clustering [Wiemerslage et al., 2021] Word-Level and Sentence-Level Canonical Morpheme Segmentation as a follow-up to 2005-2010 Morpho Challenges. [Batsuren et al., 2022]

In this thesis, we evaluate our unsupervised models by using proposed challenges on the following morphological tasks:

(1) Surface-level morphological segmentation: A model needs to segment the

word into its morphemes. For example, in Turkish "evlerimizdeki" (in our houses) should be segmented as *ev +ler+imiz+de+ki*.

(2) Morphological reinflection: A model needs to reinflect the given the word with provided morphological features and construct the correct surface form. The task, therefore, can be thought of as two-step; in the first step, the model needs to identify the lemma of a word, then it should apply the inflection with the given morphological features. As a Spanish example, santificar (to sanctify) with morphological features V; IMP; PL;2; NEG should be inflected as "no santifiquéis" (do not sanctify) [Cotterell et al., 2016a]. During training, unsupervised models are not explicitly given any morphological tags; they observe raw surface forms. Therefore, as part of the reinflection task, we train our model under semi-supervision by providing some data with its morphological tags, whereas the rest is not. We compare our results with training only on supervised data to show how unsupervised data contribute to our results.

(3) Probing morphological features: Classifying model representations for morphological features. A latent space of models is established for two different morphological features: person and tense. Using the dataset, we first pretrain models. Our next step is to train a linear classifier based on the representations. Having high accuracy is interpreted as knowing related morphological information.

The following chapters are organized as follows:

- **Chapter 2 - Models** : explains the architectures of unsupervised models and pre-training strategies.
- **Chapter 3 - Evaluations: Experiments & Results**: explains the morphological tasks, datasets and results.
- **Chapter 4 - Related work**: explains the prior work on the evaluation tasks.
- **Chapter 5- Conclusion**: summarizes the work and mentions future work.

# Chapter 2

# MODELS

## *2.1 Vector-Quantized Variational Autoencoders*

Vector-Quantized Variational Autoencoder (VQVAE) [Van Den Oord et al., 2017] is a generative model with discrete latent variables originally proposed for images and speech data. It has an encoder-decoder architecture, where the output vector of the encoder is quantized through a codebook having a number of entries. The quantization is done via selecting the minimum squared $l_2$ loss between the latent vector and codebook embeddings. After that, selected quantized vectors are fed to the decoder, and the reconstruction of original data is aimed.

Although the original model achieves promising results on image and speech data, it has not been well studied for text data. In language, we produce the words by combining a finite number of morphological features. Therefore we can think these features are in discrete space. With this motivation, we propose to increase the number of codebooks in the latent space.

We construct a VQVAE model with continuous and discrete variables with the following blocks: bidirectional LSTM encoder, low-dimensional continuous space, multiple codebooks for discrete space, and a unidirectional LSTM decoder. While the continuous part is regulated by KL divergence to standard Gaussian prior as in regular VAE, the discrete latent variables are obtained with quantization through multiple codebooks. The encoder $q$ with parameters $\phi$, has last forward hidden state $\overrightarrow{h_t}$ and last backward hidden state $\overleftarrow{h_t}$ with $d$ dimensional vectors. Applying linear transformation to $\overrightarrow{h_t}$ the mean $\mu$ and variance $\sigma^2$ are learned. Then using $\mu$ and $\sigma^2$, we estimate the continuous latent variable $z_c$. To make learning step differentiable, we use reparametrization trick [Kingma and Welling, 2013] and calculate $z_c = \mu_\phi(x) + \sigma_\phi(x) * \epsilon$ where $\epsilon \sim N(0, 1)$.

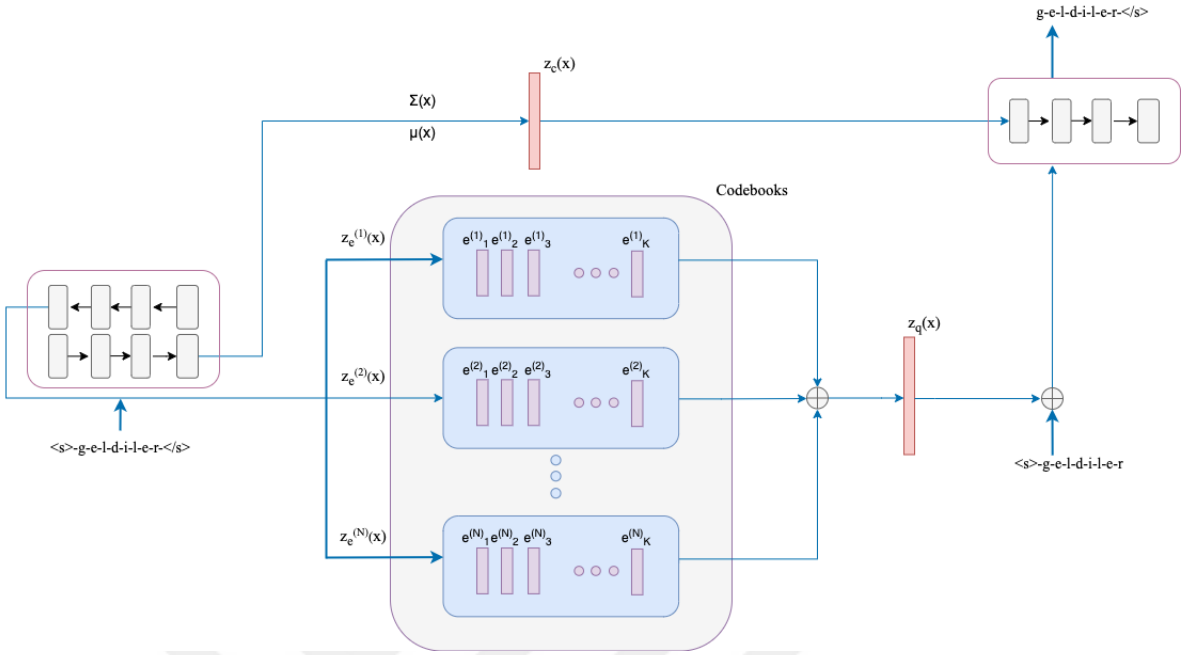


Figure 2.1: Multiple codebook VQVAE with continuous and discrete latent variables

For quantization, we define the latent embedding space of codebooks as $e \in R^{N \times K \times D}$ where $N$ is the number of codebooks, $K$ is the number of entries in each codebook, $D$ is the dimension of each embedding vector $e^{(n)}$ which is equal to $dim(\overleftarrow{h_t})/N$. The $\overleftarrow{h_t}$ vector from encoder is then divided into $N$ vectors with dimension $D$. For each divided vector from encoder $z_e^{(n)}(x)$, the nearest embedding from $codebook^{(n)}$ is calculated:

$$q(z_q^{(n)} = k|x) = \begin{cases} 1 & for\ k = argmin_j ||z_e^{(n)}(x) - e_j^{(n)}||_2 \\ 0 & otherwise \end{cases} \tag{2.1}$$

Then we concatenate the quantized vectors $z_q^{(1)}, z_q^{(2)}, ...z_q^{(N)}$ and obtain $z_q(x)$ vector.

The total objective for our model becomes:

$$
\begin{aligned}
L = \; & \mathbb{E}_{z_c \sim q_\phi(z|x)}[\log p(x|z_c, z_q)] \\
& + \sum_{n=1}^{N} ||sg[z_e^{(n)}(x)] - e^{(n)}||_2^2 + \beta||z_e^{(n)}(x) - sg[e^{(n)}]||_2^2 \\
& - KL(q(z_c|x)||p(z_c))\}
\end{aligned} \tag{2.2}
$$

The first part of the loss is the reconstruction loss, where the model conditions both continuous latent variable $z_c$ and discrete latent variable $z_q$ to reconstruct the observed data x. The second term is the total vector quantization loss for each vector $z_e^{(n)}(x)$. As in the original VQVAE, to learn the dictionary embeddings $e^{(n)}$, stop gradient operation (which is denoted as sg) is used. Stop-gradient makes the gradient of applied term zero in forward computation time and converts it to a non-updated constant. In minimizing $l_2$ distance, only the dictionary embeddings are updated. The same is applied to update the encoder outputs $z_e^{(n)}(x)$ to minimize the $l_2$ which is called *commitment loss* weighted by parameter $\beta$ to ensure encoder outputs do not grow faster than dictionary embeddings. Lastly, the last term regulates the continuous vector with a Gaussian distribution. When defining a uniform prior on $K$ categories, a constant KL divergence $\log K$ can be obtained, and since it is constant, it can be ignored.

### *2.1.1 Posterior Collapse*

The model is susceptible to posterior collapse for both continuous and discrete latent variables. Regularizing $z_c$ with KL divergence may result in a latent variable $z_c$ that will be close to prior but during decoding model only conditions on the last prediction of decoder like language models without taking into account the latent variable $z_c$. To avoid this issue, we use the KL annealing strategy where the KL divergence term in loss (2.2) is weighted with $\lambda$, which begins with zero, and gradually increases until a predefined threshold $\lambda_t$. It is expected that at the early stages of regularization, the model first learns to project the input in a low-dimensional space. At later stages, the regularization will omit unnecessary information from the projection.

### *2.1.2 Separation of Lemma and Suffixes*

Our primary goal in designing this model is to separate the lemma and suffixes of words using continuous and discrete latent variables. It is possible, however, that a learned model will not meet these expectations. In one scenario, both lemma and suffix information can be encoded in continuous parts, and during decoding, the model may only use the continuous latent variable to reconstruct the data. Alternatively, the model may encode lemma and suffix information into codebook embeddings and only use the discrete latent variables during reconstruction. Hence, besides evaluating model reconstruction accuracy, we also evaluate lemma and suffix separation during training.

We quantitatively observe the separation of lemma and suffixes using the probing technique. Probing involves freezing the encoder parameters, attaching linear probes to continuous and quantized parts of latent variables, and then predicting suffix or lemma information within those representations. For probing experiments, we filter the training and test sets of SIGMORPHON 2016 shared task-3 for Turkish and Hungarian, which includes labeled data for probed morphological tags.

Table 2.1: Probe accuracies on the Turkish test set of continuous latent variable $z_c(x)$ and quantized latent variable $z_q(x)$ of VQ-VAE. A random baseline is calculated with predictions of a uniform random variable over the probe categories.

| **Probe** | **#Train data** | **#Test data** | **Baseline** | $z_c(x)$ | $z_q(x)$ |
|---|---|---|---|---|---|
| Tense | 2791 | 375 | 0.34 | 0.45 | 0.97 |
| Person | 2824 | 383 | 0.34 | 0.35 | 0.94 |
| Number | 12681 | 1586 | 0.50 | 0.56 | 0.91 |
| Poss | 7963 | 964 | 0.16 | 0.17 | 0.79 |
| Case | 9856 | 1204 | 0.16 | 0.20 | 0.95 |
| Aspect | 1924 | 264 | 0.25 | 0.34 | 0.98 |
| Lemma | 12336 | 1597 | $\approx 0.01$ | 0.84 | 0.23 |

The probing results Table 2.2 demonstrate that, while quantized variables dominate morphological tag information, continuous variable $z_c$ probes have closer scores

Table 2.2: Probe accuracies on the Hungarian test set of continuous latent variable $z_c(x)$ and quantized latent variable $z_q(x)$ of VQ-VAE. A random baseline is calculated with predictions of a uniform random variable over the probe categories.

| **Probe** | **#Train data** | **#Test data** | **Baseline** | $z_c(x)$ | $z_q(x)$ |
|---|---|---|---|---|---|
| Tense | 14877 | 1000 | 0.65 | 0.66 | 0.99 |
| Person | 15419 | 1000 | 0.56 | 0.55 | 0.99 |
| Case | 969 | 143 | 0.09 | 0.02 | 0.99 |
| Mood | 13886 | 1000 | 0.54 | 0.56 | 0.96 |
| Def | 13386 | 1000 | 0.57 | 0.57 | 0.93 |
| Lemma | 16219 | 2344 | $\approx 0.005$ | 0.97 | 0.24 |

to the baseline, which is consistent with our intuition that suffix-related information will be concentrated in discrete variables. However, even though most of the lemma information appears to be continuous, there is still considerable information about the lemma presented in quantized form. This could indicate that some lemma information persists in a discrete form. Further probing experiments are also applied in section 3.1.

## *2.2 Baselines*

### *2.2.1 Character-Level Language Models*

Character-level language models are generative models that learn to assign probabilities to the sequence of characters in the language (e.g., 'how likely to see characters sequence *abc* in a language?'). Each character's probability is predicted and conditioned on the preceding characters. Using the chain rule principle, for a character sequence with length $t$, the probability is calculated as:

$$P(w_{1:t}) = P(w_t|w_{1:t-1})...P(w_4|w_{1:3})P(w_3|w_{1:2})P(w_2|w_1)P(w_1) \tag{2.3}$$

The training objective for observed data $\mathbf{x}$ is maximizing the log-likelihood with model parameters $\theta$:

$$L = \log p_\theta(\mathbf{x}) \tag{2.4}$$

### *2.2.2 Autoencoders*

Autoencoder (AE) [Hinton and Salakhutdinov, 2006] is an encoder-decoder architecture where observed data is given as input to the model. While the encoder (a parametric function $q$) encodes the observed data into a low dimensional hidden vector $z$, the decoder (a parametric function $p$) reconstructs the data again by conditioning this vector. The training objective for observed data $\mathbf{x}$ is maximizing the log-likelihood:

$$L = \log p_\theta(\mathbf{x}|\mathbf{z}) \tag{2.5}$$

In the word case, it can be argued that the encoded vector includes all the necessary information to construct a word's surface form. Thus, one can simply analyze the vector produced by an encoder to understand how the model captures morphological information.

### *2.2.3 Variational Autoencoders*

Variational Autoencoder (VAE) [Kingma and Welling, 2013] is a probabilistic generative model. While VAE is an architecture similar to AE, its principle is to learn a distribution over the latent variables of observed data. It uses two parametric functions $q$ and $p$, with parameters $\theta$ and $\phi$. With function $p(x|z)$ it learns the generation of data given latent variable z, with function $q(z|x)$ it learns to estimate the latent variable z for the observed data. The training objective for observed data $\mathbf{x}$ is maximizing the log-likelihood:

$$L = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - KL[q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})] \tag{2.6}$$

The first part of the loss is the reconstruction loss, where the model conditions continuous latent variable $z$ to reconstruct the observed data x. The second term regulates the continuous vector with a prior distribution. As a prior, it is generally assumed multivariate standard Gaussian distribution. At the end of its training, the VAE encoder can encode the observed data into mean-shifted Gaussian vectors. The decoder can generate data by conditioning that vector. Therefore, at the generation step, one can sample a vector from prior and feed it to the decoder. Another generation method can be interpolation between the latent variables of observed data. In order to understand capturing morphological information at VAEs, the encoded latent vector is a suitable candidate as in autoencoders.

# Chapter 3

# EVALUATIONS: EXPERIMENTS & RESULTS

## *3.1 Probing*

Probing is a method for interpreting neural models' representations and identifying relevant information. A neural model encodes input data into different representations throughout its layers. Often, these representations are high-dimensional, which makes interpreting them difficult. Interpretability is achieved through the use of classifiers in probing. Adding a classifier over a model's representations can help us determine if these representations are classified similarly to the way humans do. Consider the following example: If a language model is given an adjective such as *yellow*, it will most likely complete it with a noun, such as *houses, bikes, shirts, etc.*. This begs whether the model can recognize nouns, verbs, and conjugations. In this case, we can obtain the model's word representations, train a probing classifier on top of them and observe if the representations are classified according to their part of speech (PoS). Then based on the results, we can conclude that the model's representations encode words PoS tag.

Probing can be applied not only for diagnosing specific linguistic properties in the model's representation but also to find out what information is being used in the model's predictions. The classification results and the model's predictions can be compared to see any causal relation between the encoded feature and the task. In order to do this, one can update the model's representation with probe gradients and see its effect on the model's behavior.

For morphological evaluation, a probing procedure can be applied to the morphological features of words. One might ask, for example, if a model encodes the grammatical cases of nouns, like the accusative noun cases -the direct objects of verbs e.g. *kitabi okuyor*, *hava alanini soruyor*, *sokağı gecti*-, or the dative cases -the

indirect objects of verbs, e.g. *kitaba not yazdi*, *hava alanina dogru ilerledi*, *yüzüğü ariyor*- in Turkish. In order to do so, one can feed a model with words, attach a *diagnostic probe* to some picked region and observe if it classifies the words according to their case features. Any other morphological feature knowledge of a model can be studied similarly. Moreover, behavioral tests can be applied to see whether the models use the morphological features diagnosed by probes. Causal probing ideas can also be applied by intervening in different parts of models' representation to determine at what level the information is being used.

Our study uses probes to diagnose the encoded features in the model's representations. We first pre-train unsupervised models with encoder-decoder architectures: AE, VAE, and VQVAE with multiple codebooks. Then we train linear classifiers on top of their low dimensional encoding vectors. We also provide visualizations of our probing results.

### *3.1.1 Model*

AE: We pre-train it with a bidirectional LSTM encoder and a unidirectional LSTM decoder with hidden sizes 512. We set the dimension of the feature vector to 32. Character embeddings are 256.

VAE: We pre-train it with a bidirectional LSTM encoder and a unidirectional LSTM decoder with hidden sizes 512. We set the dimension of mean and standard deviation vectors to 32. Character embeddings are 256.

VQVAE: We use four codebooks, each of which has five entries. We pretrain it with a bidirectional LSTM encoder with a hidden size of 288 and a unidirectional LSTM decoder with a hidden size of 256. We set the low dimension to 32. The prior for KL regularization of continuous latent variables are standard multivariate Gaussian. Character embeddings are 128. The hidden forward states are encoded in low-dimensional continuous vectors, and the backward hidden states are quantized.

### *3.1.2 Diagnostic Probing*

In diagnostic probing, we examine the latent space of models for two different morphological features: tense and person. We first pretrain the models on the dataset. After that, we freeze model parameters and train a linear classifier on top of representations. High accuracy is interpreted as knowledge of related morphological knowledge.

- For AE, we probe the feature vectors, which are the reduction of encoder outputs (32-dimensional).
- For VAE, we probe the mean vectors, which are the reduction of encoder outputs (32-dimensional).
- For VQVAE with multiple dictionaries, we probe the continuous mean vectors, which are the reduction of encoder outputs (32-dimensional) and the concatenation of codebook outputs (288 dimensional).

| **Tense** | **# of words** |
|---|---|
| Aorist | 690 |
| Progressive | 994 |
| Past | 1197 |
| Future | 432 |
| Narrative | 383 |
| **Total** | 3696 |

Table 3.1: Tense data statistics for pretraining and probing.

| **Model** | **Accuracy** |
|---|---|
| AE | 0.97 |
| VAE | 0.99 |
| VQVAE (cont.) | 0.41 |
| VQVAE (quant.) | 0.85 |
| **Baseline** | 0.32 |

Table 3.2: Probing accuracies of models. The baseline is the percentage of majority tense (progressive1) in the dataset. While both AE and VAE representations are classified well, the VQVAE continuous latent vector is classified with a slightly better than baseline, indicating that the morphological features are primarily represented in quantized variables of codebooks as we have intended.

*Tense Probe*

For tense probing, we use a Turkish dataset containing 3696 verb words with five tense classes: Aorist, Progressive1, Past, Future, and Narrative.

| **Person** | **# of words** |
|---|---|
| A1sg | 693 |
| A2sg | 179 |
| A3sg | 1708 |
| A1pl | 358 |
| A2pl | 296 |
| A3pl | 462 |
| **Total** | 3696 |

Table 3.3: Person tag data statistics for pretraining and probing.

| **Model** | **Accuracy** |
|---|---|
| AE | 0.99 |
| VAE | 0.98 |
| VQVAE (cont.) | 0.47 |
| VQVAE (quant.) | 0.83 |
| **Baseline** | 0.46 |

Table 3.4: Probing accuracies of models. The baseline is the percentage of the majority person (A3sg) in the dataset.

*Person Probe*

We use a Turkish dataset containing 3696 verb words for person probing with six person tag classes: A1sg, A2sg, A3sg, A1pl, A2pl, and A3pl.

### *3.1.3 Visualizations with Probes*

We can also clearly visualize how the model's hidden space is transformed using probes.

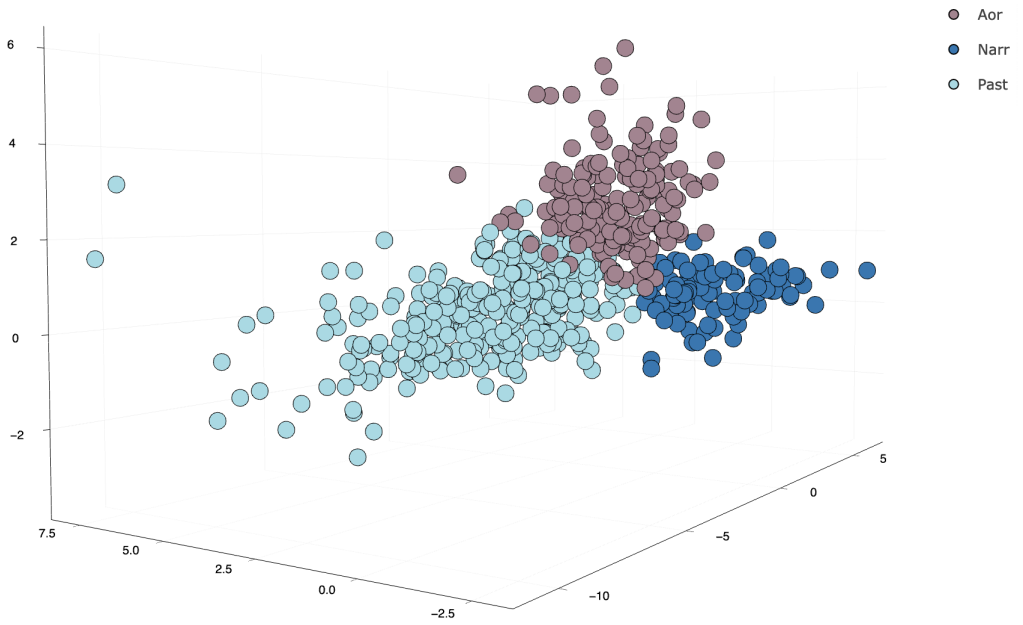


Figure 3.1: Tense probing scores of VAE mean vectors

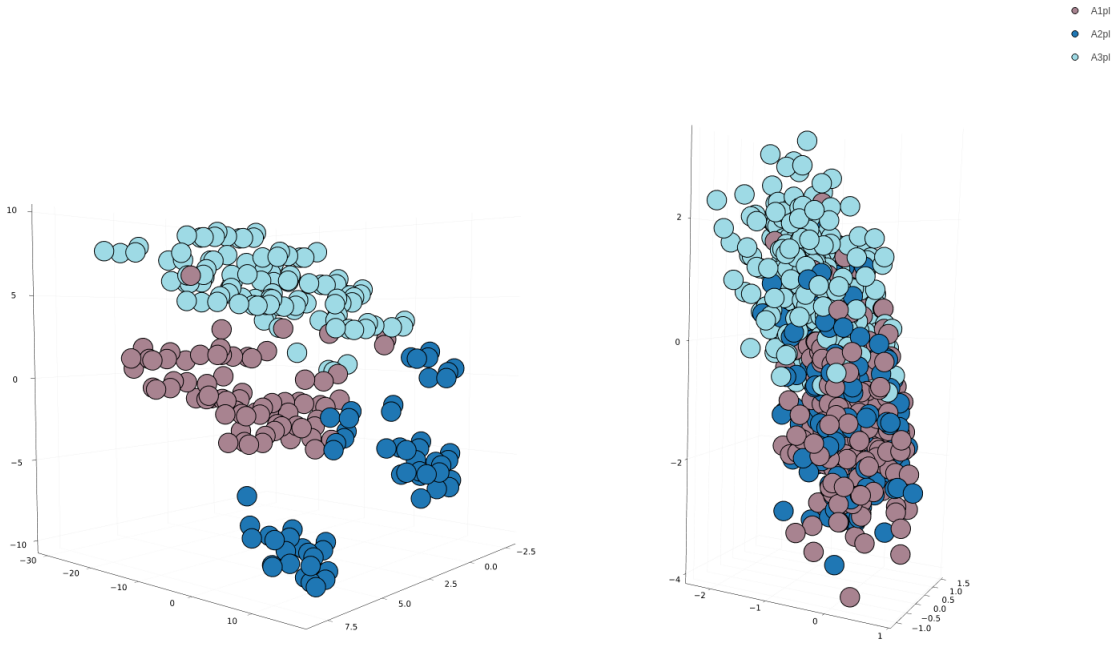


Figure 3.2: Person probing scores of VQVAE. Left: Probes on quantized latent representation. Right: Probes on continuous representations. While quantized latent representations cluster well, continuous representations are more mixed.

### *3.1.4 Results*

In this section, we trained probing classifiers for two different morphological features, tense and person of Turkish words, on low-dimensional AE, VAE, and VQVAE representations. We observed that while for a person and tense features, AE and VAE can be classified almost perfectly. For VQVAE, we observed that the morphological feature information was encoded in both continuous and discrete parts, but primarily discrete.

### *3.2 Morphological Segmentation*

A morphological segmentation task requires the model to segment words into morphemes. Word-level morphological segmentation can be done in two ways: canonical segmentation or surface segmentation. In surface segmentation, the word $w$ is divided into its surface morphemes $s$, which form the word's orthography. For example, the word *comparability* has a surface segmentation as *compar+abil+ity*. On the other hand, in canonical segmentation, the word is split into its *canonical* morphemes which should be *compare+able+ity*. Therefore, the canonical morpheme $c$ of a word is a modified version of a surface morpheme $s$ of a word. Another example from German, the word *internationalisierung* (international) which has surface segmentation as *international+isier+ung*, and canonical segmentation as *internationale+isier+ung*.

For accurate surface-level segmentation, each morpheme must maintain its surface form when forming a word. For example, in fusional languages, during word formation, the surface form of morphemes changes [Mager et al., 2020]. Also, different morphological processes must be considered during inflection for agglutinative languages such as Turkish or Finnish. For example in Turkish, vowel and consonant deletions, *ufak+cık* → *ufacık* or consonant softening *çiçek+i* → *çiçeği* . In those cases, surface-level segmentation produces subwords that are not meaningful, which violates the definition of morpheme that they are the smallest units that bear meaning in a word. There is also the problem of ambiguous cases. For example, a word *yazımda* can be both understood as "in my writing" and needs to be segmented as *yazı-m-da*, or "in my summer" *yaz-ı-mda* or "in the writing" and needs to be segmented as *yazım-da*.

For unsupervised models, both segmentation methods can be studied with various heuristics. In this section, we study Turkish surface-level segmentation with our unsupervised models.

### *3.2.1 Model*

The task requires identifying morpheme boundaries within words. At every morpheme boundary, a word gains validity; it becomes meaningful since morphemes are its smallest meaningful units. It is, therefore, possible to use a model that can detect meaningful/meaningless words for morphological segmentation. Probabilistic generative models achieve this by learning the distribution of words in a dataset. They assign high probabilities to valid words but low probabilities to invalid or incomplete words. Therefore they are suitable for the segmentation task. In this section, we experiment with two of them: VAE and CharLMs.

For VAE, we use a bidirectional LSTM encoder and a unidirectional LSTM decoders with a hidden size of 512. The latent variable dimensions, mean and standard deviation vectors reduced from encoder output, are set to 32. The embedding size for characters is 256. We use the Adam optimizer with a learning rate of 0.001. To prevent posterior collapse, we apply the KL annealing strategy: The KL divergence term in loss is weighted with a parameter that begins with zero and gradually increases until the threshold of 0.2. For CharLM, we use an LSTM decoder with a hidden size of 512. The embedding size for characters is 256. We use Adam optimizer with a learning rate of 0.001.

After pretraining, to segment a word $w$, we ask the model to evaluate the likelihood of each subword within itself. For example for the word *yolda*, the subwords are *y‹/eos›*, *yo ‹/eos›*, *yol‹/eos›*, *yold‹/eos›*, *yolda‹/eos›*. We then apply the following heuristics to those probability values:

**Heuristic** We first define the subwords of word $w_{1:n}$ as $w_{1:1}$, $w_{1:2}$ ... , $w_{1:n-1}$, $w_{1:n}$. Then we list the likelihoods of each subword within the word. When the likelihood of a subword exceeds the likelihood of the previous and next subword, we consider this morpheme boundary. That is, we choose the endpoint of subword $w_{1:i}$ as a morpheme boundary if:

- $\mathrm{p}(w_i) > \mathrm{p}(w_{i-1})$ and,

- p($w_i$) > p($w_{i+1}$) and,
- The length of $w_i > 2$

Since our models learn to assign probabilities to the sequence of characters in the given language, they should detect when incomplete words end with an end-of-sentence (eos) marker, assigning lower probabilities for invalid/incomplete words and higher probabilities to valid/complete words. At morpheme boundaries, the words are valid. Therefore whenever a subword $w_i$ is at the point of boundary, the assigned probability $p(w_i)$ will be high. However, one character before the boundary, the word $w_{i-1}$, will probably be incomplete; therefore, the assigned probability $p(w_{i-1})$ will be low. Again, one character after the morpheme boundary, the word $w_{i+1}$, will probably be incomplete, and the assigned probability $p(w_{i+t})$ will be lower. For example, a word *sordular* ("they asked" in English) should be segmented as *sor-du-lar*, the words *so*, *sord*, *sordul*, *sordula* are invalid therefore have lower probabilities than the boundaries *sor*, *sordu* and *sordular*.

To apply our heuristics, we need to obtain subword likelihood $p(w_i)$ from models. For VAE, it can be estimated using importance sampling:

$$p(w_i) = \mathbb{E}_{z_n \sim Q(z_n)} \frac{p(w_i|z_n)p(z_n)}{q(z_n|w_i)} \tag{3.1}$$

$$p(w_i) \approx \frac{1}{N} \sum_{n=1}^{N} \frac{p(w_i|z_n)p(z_n)}{q(z_n|w_i)} \tag{3.2}$$

$$\log p(w_i) \approx -\log N + \log \sum_{n=1}^{N} \frac{p(w_i|z_n)p(z_n)}{q(z_n|w_i)} \tag{3.3}$$

To determine the number of samples $N$, we apply a grid search between 8 and 128000 samples. For each $N$, we run the experiments five times and calculate the mean and standard deviation of the marginal likelihood of randomly 50 words, including subwords. As seen in fig 3.2.1, for most words, the results converge. To be

computationally efficient, we pick sample size $N$ as 32000.

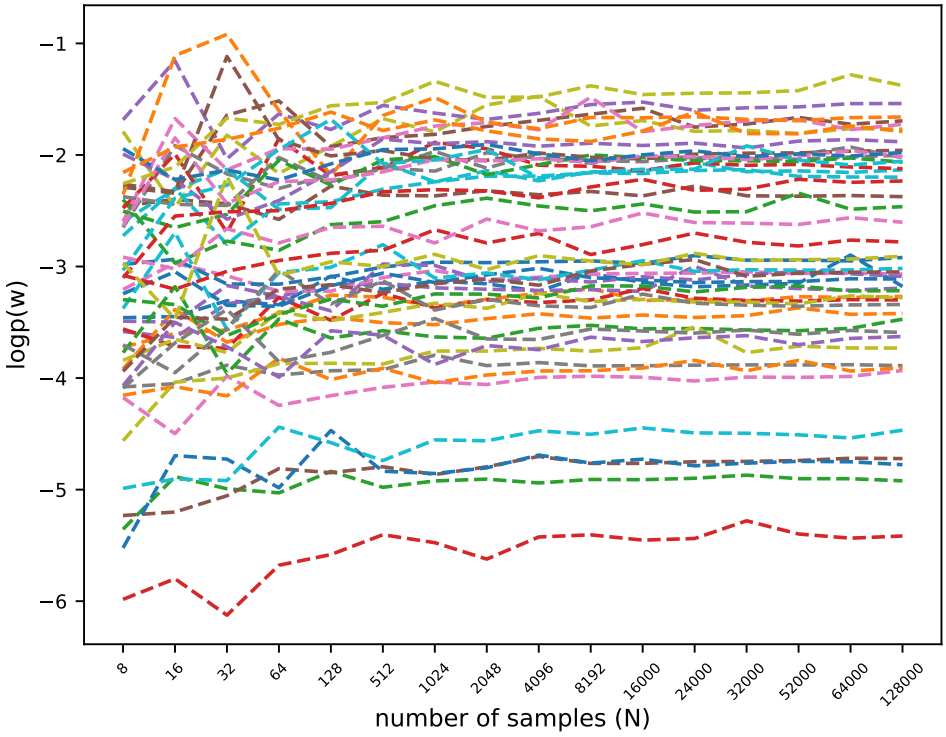


Figure 3.3: Marginal likelihood estimations of VAE for randomly selected 50 words from the dataset.

For CharLM, word likelihood $p(w_i)$ is simply:

$$\log p(w_i) = \sum_i \log p(w_i|w_{i-1}, ..., w_0) \tag{3.4}$$

### *3.2.2 Data & Experiments*

We experiment on Turkish and use Morpho-Challenge 2010 dataset, including 617k training words. We filter the training data with the top 50k frequent data. For test sets, we follow the MorphoChain [Narasimhan et al., 2015] and aggregate the dev and test sets from 2005-2010 years. We additionally provide experimental results on Hungarian in the Appendix.

| **Language** | **Train (# of words)** | **Test (# of words)** |
|---|---|---|
| Turkish | 50000 | 2534 |

Table 3.5: Data statistics for morphological segmentation

| **Model** | **Precision** | **Recall** | **F1** |
|---|---|---|---|
| LSTM-Random | 36.54% | 33.65% | 35.03% |
| *CharLM | 66.10% | 65.75% | 65.93% |
| *VAE | 64.33% | 66.45% | 65.38% |
| Morphagram | 84.11% | 63.60% | **72.43**% |
| Morfessor EM+Prune | 74.38% | 46.82% | 57.47% |
| MorphoChain | 74.30% | 52.00% | 61.20% |

Table 3.6: Morphological segmentation results on the test set. *:Our models

*Baselines*

We compare our results with other unsupervised methods Morfessor EM+Prune [Grönroos et al., 2020], MorphoChain [Narasimhan et al., 2015] and Morphagram [Eskander et al., 2020]. We report MorphoChain results from their paper; for the others, we train the models from scratch.

### *3.2.3 Error Analysis*

In that section, we specifically analyze VAE, categorize the errors in three as *under segmentation*, *over segmentation* and *wrong segmentation*. Specifically, we randomly sampled 200 words that were segmented incorrectly and analyzed them.

Over 200 sampled words; we see that 39.5% under segmentation, 43.0% over-segmentation, 14.0% wrong segmentation and 0.5% are dataset annotation mistakes.

The most common error pattern for under-segmentation cases is about the consecutive valid subwords. For example, the word *uzayInda* (at someone's universe) should be segmented as *uzay-In-da*. However, since *uzay*, *uzayI*, *uzayIn* are all valid words and since our model assigns probabilities $p(uzay) < p(uzayI) < p(uzayIn)$,

our heuristic can only identify boundary between *uzayIn* and *uzayInd*, therefore segments the word as *uzayIn-da*. This issue also disables the identification of one-letter-morphemes since words, including one-letter-morpheme, always have two consecutive valid subwords. For example, assume the case of word *tanIrlar* which should be segmented as *tanI-r-lar*. Again, since *tanI* and *tanIr* are both valid word forms, and the model assigns $p(tanI) < p(tanIr)$ we miss the correct boundary, and segment the word only as *tanIr-lar*.

For over-segmentation cases, we observe that 55% of them occur at the stem, and 45% of them occur at the suffix. The common reason behind over-segmentation is that a subword is a valid word, but it is not the correct morpheme boundary in the context of the word. For example, the word *istedin* (you wanted) should be segmented as *iste-din*. However our model instead splits as *iste-di-n* since *istedi* is also a valid word in Turkish (he/she wanted). In that case if model assigns $p(istedi) > p(istedin)$ the over-segmentation occurs.

For wrong segmentation cases, where the issue is not only missing or having additional morpheme boundaries but also inconsistent boundaries, such as *paramIza* that should be segmented as *para-mIz-a* but instead segmented as *param-Iza*, the first reason is also common with under and over-segmentation cases: a word that includes two consecutive valid subwords. For example, in case of word *Ucretimi* which should be segmented as *Ucret-im-i*, the model segments it as *Ucreti-mi* with following error sequence: it first misses the morpheme boundary between *Ucret* and *Ucret-i* because of probability assignments $p(Ucret) < p(Ucreti)$. Second, it produces an additional boundary between *Ucreti* and *Ucretim* with $p(Ucreti) > p(Ucretim)$. Finally, it misses the last morpheme boundary between *Ucretim* and *Ucretimi*, because of $p(Ucretim) < p(Ucretimi)$. In the end, the boundaries are detected inconsistently.

Also, another reason for wrong segmentation is consonant softening and consonant deletion cases. As a consonant softening example, the word *erkeGini* has the root *erkek*, but the letter $k$ changes to letter $G$ during inflection because of the consonant softening rules. In that case, the gold segmentation in the test dataset is *erkeG+i+ni*. However, it is unclear that it should be the case since the word *erkeG* is not meaningful. As a vowel deletion example, the word *keSfinin* has the root *keSif*

| **word** | **gold segmentation** | **predicted segmentation** |
|---|---|---|
| tanIrlar | tanI-r-lar | tanIr-lar |
| puanInda | puan-In-da | puanIn-da |
| kIyImInI | kIy-Im-I-nI | kIyIm-InI |
| uyaranIn | uyar-an-In | uyaran-In |
| sevilmiS | sev-il-miS | sevil-miS |

Table 3.7: (VAE) Under segmentation cases. Forms 39.5% of errors.

| **word** | **gold segmentation** | **predicted segmentation** | **part** |
|---|---|---|---|
| buluttan | bulut-tan | bulut-ta-n | suffix |
| yeltendi | yelten-di | yel-ten-di | stem |
| plakete | plaket-e | plak-et-e | stem |
| kanmaz | kan-maz | kan-ma-z | suffix |
| yazdır | yaz-dır | yaz-dı-r | suffix |

Table 3.8: (VAE) Over segmentation cases. Forms 43% of errors.

| **word** | **gold segmentation** | **predicted segmentation** |
|---|---|---|
| Ucretimi | Ucret-im-i | Ucreti-mi |
| canbazIn | canbaz-In | can-ba-zIn |
| devSiren | devSir-en | dev-Si-re-n |
| keSfinin | keSf-i-nin | keSfi-nin |
| paramIza | para-mIz-a | param-Iza |

Table 3.9: (VAE) Wrong segmentation cases. Forms 14% of errors.

| **word** | **gold** | ***VAE** | ***CharLM** | **MorphAGram** |
|---|---|---|---|---|
| buluttan | bulut-tan | bulut-ta-n | bulut-tan | bulut-tan |
| yeltendi | yelten-di | yel-ten-di | yel-ten-di | yelten-di |
| plakete | plaket-e | plak-et-e | plaket-e | plaket-e |
| kanmaz | kan-maz | kan-ma-z | kan-ma-z | kanmaz |
| yazdır | yaz-dIr | yaz-dI-r | yaz-dI-r | yaz-dIr |
| sOndUrmeye | sOn-dUr-me-ye | sOn-dUr-me-ye | sOn-dUr-me-ye | sOn-dUrme- ye |

Table 3.10: Segmentation among different models. While VAE and CharLM(LSTM) have over-segmentations for different words, MorphAGram can handle over-segmentations. However it suffers from under-segmentations.*:our models.

but during inflection the vowel *i* is deleted. In that case, the gold segmentation is expected to be *keSf+i+nin*. Again, it is a vague case of what should be the stem. These consonant cases indicate surface segmentation's drawbacks in a language such as Turkish.

As a better model than ours, we also evaluate the results from MorpAGram [Eskander et al., 2020]. We first train it following the pipeline and hyperparameters defined in the paper. MorphAGram uses transductive learning; the test set instances are also contained in the training set. Therefore we first merge the training set and test set. As seen in table 3.2.2, MorphAGram has much higher precision and lower recall than our models. We see that most of our over-segmentation failures are handled by MorphAGram. (Table 3.2.3) However, it also suffers from under-segmentation, as its recall score indicates.

### *3.2.4 Results*

As a result, most of the errors we observed in this chapter could be considered weaknesses of our heuristic. The models, however, can distinguish correct from incorrect surface forms, allowing them to be used for surface-level segmentation. We performed experiments on Turkish, which is highly agglutinative; the word's stem is identified at first morpheme boundary detection, and the suffixes are identified at the following boundary detections. For languages with affixations, the probability assignments of generative models can also be combined with different heuristics.

### *3.3 Morphological Reinflection*

Morphological reinflection is the task of studying the relation between inflections of words and how much a model can generalize the inflection rules of words. In a language, inflected forms of words are related to each other. For example, think about two verbs from Turkish *kalmak* (to stay) and *uyanmak* (to wake up) and some of their inflected forms. To indicate positive+progressive+$1^{st}$ person singular, the inflections will be *kalıyorum-uyanıyorum*. To indicate positive+narrative+$1^{st}$ person singular, the inflections will be *kalmış-uyanmış*, to indicate negative+future+$3^{rd}$ person plural, they will be *kalmayacaklar-uyanmayacaklar* and so on. After that time, if we know that the positive+past+ $1^{st}$ person for verb *kalmak* is *kaldım*, we can generalize and say that for the word *uyanmak* the positive+past+ $1^{st}$ person will be *uyandım*. While they have different roots, they are inflected similarly to convey the meaning. Another example from the noun words *fincan* and *cam*. To indicate accusative case: their inflected forms will be *fincanı-camı*, for dative case, *fincana-cama*, for ablative case *fincandan-camdan* and so on. Then if we know that the locative case for fincan is *fincanda*, we will guess that for *cam* the locative case will be *camda*.

In order to able to accomplish the task, different morphological processes are needed to be learned. For example in Turkish, to inflect the nouns to plural forms, the first rule is vowel harmony, where if the last vowel is *a,ı,o,u* the suffix should be followed by *a* such as *okul-lar*, if it is *e,i,ö,ü* the suffix should be followed by *e*, such as *gül-ler*. Alternatively, if the stem ends with a vowel, the first vowel of the morpheme is deleted during inflection. For example, for possessive morpheme *-I'm,-İm*, *masa+Im* → *masam*, *oda+Im* → *odam*, or for progressive tense morpheme *-Iyor, -İyor* surface forms will be generated by deleting the last vowel from the verb *bağla+Iyor* → *bağlIyor*. Another rule that is needed to be learned is consonant devoicing such as *maç+da* → *maçta*, or consonant changes such as *yaprak+ı* → *yaprağı*. In summary, many rules exist to inflect the words in the correct surface form.

As stated in [Cotterell et al., 2016b], the relation between inflected forms of

words allows us to generate and analyze the words in a language, even though the language may include many combinations of inflections. We can learn the shape and the way of suffixation from some examples and infer it for other words. The task is also crucial to handle data sparsity when a model does not see every inflected form of a word but can generalize to unseen words by learning the inflection rules.

In the task, a model is given an inflected word form $, w_s$ and a sequence of morphological tags $y_t$ and is asked to predict the reinflected word $w_t$. For example, for an inflected word *gelecekler* with tags negative+past+$1^{st}$ person plural, the model needs to output *gelmedik*. Unsupervised models, however, during training, are not explicitly given any morphological tags; they only observe the raw surface forms of words. Therefore, to accomplish the reinflection task with an unsupervised model, we need to find a way to indicate the asked morphological tags to the model.

In this chapter, we train our model with semi-supervision by giving some part of the data with its morphological tags, whereas the rest is given without any tags information. To show the contribution of unsupervised data, we compare our results with training only on supervised data.

### *3.3.1 Model*

For each language, we use VQVAE with the number of codebooks and entries with the number of morphological features in the language dataset. (Turkish:11, Spanish:11, Hungarian:10, Georgian:8) We use a bidirectional LSTM encoder with a hidden size of (Turkish:1320, Hungarian:660, Spanish:660, Georgian:640) and a unidirectional LSTM decoder with a hidden size of 512 for each language. The latent dimension for mean and standard deviation vectors is set to 128. The embedding size for characters is 256. We use the Adam optimizer with a learning rate of 0.001. The KL divergence term in loss is weighted with a parameter that begins with zero and gradually increases until the threshold of 0.2.

### *3.3.2 Data & Experiments*

We use SIGMORPHON 2016 Shared Task Data on Morphological Inflection and experiments on four languages: Turkish, Hungarian, Spanish, and Georgian. To train our model, we follow [Zhou and Neubig, 2017] and gather unlabeled data from task-1 and task-2 training and validation sets as well as task-3 training data.

*Semi-supervision for codebook entry selection*

We supervise the dictionary selection with the morphological tags during the model's training. In that case, the objective for the model in 2.2 is slightly changed to:

$$\begin{aligned} L_{sup} = \; & \mathbb{E}_{z_c \sim q_\phi(z|x)}[\log p(x|z_c, z_q)] \\ & + \sum_{n=1}^{N} ||sg[z_e^{(n)}(x)] - e_{j*}^{(n)}||_2^2 + \beta||z_e^{(n)}(x) - sg[e_{j*}^{(n)}]||_2^2 \\ & - KL(q(z_c|x)||p(z_c))\} \end{aligned} \tag{3.5}$$

where $j*$ indicates the gold label entry for the $n^{th}$ morphologic feature.

$$\begin{aligned} L_{unsup} = \; & \mathbb{E}_{z_c \sim q_\phi(z|x)}[\log p(x|z_c, z_q)] \\ & + \sum_{n=1}^{N} ||sg[z_e^{(n)}(x)] - e^{(n)}||_2^2 + \beta||z_e^{(n)}(x) - sg[e^{(n)}]||_2^2 \\ & - KL(q(z_c|x)||p(z_c))\} \end{aligned} \tag{3.6}$$

Overall loss for our semi-supervised setting is:

$$L_{total} = L_{sup} + \alpha L_{unsup} \tag{3.7}$$

We empirically set the $\alpha$ to 0.7 for all languages.

*Baselines*

We compare our results with [Zhou and Neubig, 2017], which uses Gumbel-Softmax as a discretization method. Though their idea is very similar to our approach, they also provide supervision for reinflection with paired data (source, tag, target); we use supervision for only (target, tag, target) pairs. They also use an attention mechanism over the tag embeddings.

Table 3.11: **ULD**: unlabeled data, **LD**: labeled data.*****: our models. The performances for highly suffixing languages such as Turkish and Hungarian are comparable to the baseline. On the other hand, unsupervised data improve performances for every language.

| **Language** | **#ULD** | **#LD** | ***VQVAE (sup)** | ***VQVAE (semi-sup)** | **Baseline** |
|---|---|---|---|---|---|
| Turkish | 29608 | 12798 | 80.20 | 95.60 | 97.25 |
| Hungarian | 34025 | 19200 | 97.20 | 98.90 | 99.16 |
| Spanish | 72151 | 12799 | 73.70 | 81.40 | 93.74 |
| Georgian | 46562 | 12795 | 78.60 | 84.80 | 95.97 |

Table 3.12: The percentage of modified lemma part during reinflection (Cotterell et al., 2016) (since some words may be inflected zero or multiple times, sums may not equal 100%)

| **Language** | **Prefix** | **Stem** | **Suffix** |
|---|---|---|---|
| Turkish | 0.21 | 1.12 | 98.75 |
| Hungarian | 0.00 | 0.08 | 99.79 |
| Spanish | 0.09 | 3.25 | 90.74 |
| Georgian | 4.46 | 0.41 | 92.47 |

### *3.3.3 Error Analysis*

In this section, we analyze our model's errors in Turkish. We first look at the semi-supervised version model's errors. We categorize errors into two: *lemma errors* and *suffix errors*. We observe 71 errors over 1600 words in the test set. We count the words inflected with correct morphological features for lemma errors but include the wrong lemma. For suffix errors, we count the vice-versa. We observe that the errors are mostly because of generating the wrong lemma while keeping the correct suffixes. The second reason is correct lemma but wrong suffixation. The rest is the dataset annotation mistakes. However, the model does not produce any nonsense words.

| **source word** | **gold reinflection** | **predicted reinflection** |
|---|---|---|
| fosfor | fosforda | fostarda |
| alfabelerin | alfabenin | alametin |
| paralelkenarlar | paralelkenarları | parabolleri |
| sülükler | sülük | sülüs |
| ahudutlarını | ahududunu | ahtapotu |
| alnımıza | alında | alanda |

Table 3.13: Type-1 Errors: correct suffixes but wrong lemmas. Forms 69.0% of errors for the semi-supervised model.

| **source word** | **gold reinflection** | **predicted reinflection** |
|---|---|---|
| asetonum | asetonunun | asetonun |
| dadı | dadısında | dadında |
| gösterimin | gösteriyi | gösteri |
| ölümünüzde | ölümümüzde | ölümüzde |
| uçmuşuz | uçmuşlar | uçtular |

Table 3.14: Type-2 Errors: correct lemmas but wrong suffixes. Forms 22.5% of errors for the semi-supervised model.

| **source word** | **gold reinflection** | **reinflection (sup.)** | **reinflection (semi-sup.)** |
|---|---|---|---|
| yutuyorlardı | yutarsınız | yutursunuz | yutarsınız |
| dışımın | dışlarda | dışıarda | dışlarda |
| doymuyorduk | doyarsın | doyrursun | doyarsın |
| doğarsınız | doğuyorlardı | doğayorlardı | doğuyorlardı |
| düşünmez | düşünüyorsun | düşünyüyorsun | düşünüyorsun |
| koklamadı | kokluyordu | koklayordu | kokluyordu |
| damatlarınıza | damadımızı | damatımızı | damadımızı |
| burunlardan | burnumuzu | burunumuzu | burunumuzu |

Table 3.15: Errors after training with only supervised data. The erroneous reinflections include invalid Turkish words because of not applying phonological rules. With the addition of unsupervised data, the model handles most of it.

When we check the errors from the only-supervised version, we observe that besides having lemma and suffix errors as in the semi-supervised version, the model also suffers from learning phonological processes such as vowel deletion. For example, in the reinflection of word *koklamadı*, the last vowel in the stem should be deleted, such as kokl**a**-uyordu. However, the model does not delete the vowel and produces a wrong surface form such as *koklayordu*. Another example is the word *yutuyorlardı*. While the model can produce the correct suffix with $2^{nd}$ person plural and present tense form, it wrongly reinflects the suffix for tense *yut-are* as *yut-ur*. In the continuation, it adds the wrong allomorph of the tag as *-sunuz* instead of *-sınız*.

### *3.3.4 Results*

As a result, in this chapter, we studied the reinflection task with VQVAE in Turkish, Hungarian, Spanish, and Georgian. We did experiments in two different training settings. First, we used labeled data to give the model the supervision of codebook-entry selection during the reconstruction of a word. We then experimented with adding unsupervised data to the training set without explicit codebook entry selection. We observed that while for each language, performances got improved, for Turkish and Hungarian, where most of the modifications occur in the suffix part of the lemma, the results are comparable to the baseline. We also performed an error analysis in Turkish and showed that additional unsupervised data helps the model handle some phonological rules, such as vowel deletion or consonant assimilation.

# Chapter 4

# RELATED WORK

## *4.1 Probing*

Although NLP models have recently improved in many areas, there is still much confusion about how they work and capture linguistic knowledge. Probing studies aim to understand models' representations for observed data and discover how they encode and capture any linguistic property. Classifiers are used to interpret representations: models are classified based on linguistic property, and high accuracy is seen as evidence of knowledge in these studies.

In the first studies, it is questioned whether static word embeddings include linguistic properties. In [Köhn, 2015], word embeddings were studied with various embedding algorithms such as Continuous Bag of Words (cbow), Skip-Gram, [Mikolov et al., 2013], GloVE [Pennington et al., 2014] to predict morphological information in multiple languages (gender, case, tense, number, etc.). Researchers found that these embeddings could work well on classification tasks so that they captured morphological and syntactic information.

The idea is also applied to hidden states of recurrent neural networks [Shi et al., 2016]. Their idea comes from the nature of machine translation tasks. In order to translate a source sentence to a target sentence, some part of syntactic knowledge should be encoded, such as the active/passive property of the sentence. Therefore they study classification through source sentence embeddings. They increase their reliability by taking autoencoders baseline and show that the voice property is not *easily* accessed in that scenario. They also examine causal relationships, in which a part of the vector encodes information by pruning it. They also study word-level hidden states and show that Part Of Speech tags are encoded. The same idea has been applied in [Conneau et al., 2018], [Adi et al., 2017], [Tenney et al., 2019],

[Hupkes and Zuidema, 2018]. It was noted that most of the efforts were focused on English and sentence level, [Sahin et al., 2020] investigates word-level capture with multiple tasks for various languages. Numerous studies use probing techniques to examine different linguistic properties, as listed in [Belinkov, 2022].

Probing studies are criticized from various perspectives: the first is regarding the reliability of the probes due to the learning of parameters for the external classifiers. Control tasks are proposed by [Hewitt and Liang, 2019] to identify probes with memoization problems. The authors design some tasks that probes can only solve by memoizing to determine whether or not the results are the success of probes, and they suggest using probes without this behavior to avoid those risks. Therefore many studies use simple probes such as linear ones.

Researchers with an information-theoretic perspective argue that the measure to check should not be classification accuracy but the ease of extraction of information. [Voita and Titov, 2020] suggests using Minimum Description Length (MDL) and found out MDL results are more informative than accuracy for probes. Their idea is that if there is any regularity/classification in representations according to some linguistic property, it should be possible to compress data according to that property. Then they measure the required bits to compress the data, concluding that fewer bits indicate property encoded much more in the model's representations. [Pimentel et al., 2020] argues that the best estimate for mutual information between model representation and linguistic property comes with complex probes. Therefore they also use non-linear probes in their studies.

Several studies also question the limitation of classifications. [Michael et al., 2020] argues that the model's latent ontologies should be understood instead of predefined classes asked in classification. They observe clustering over logits by applying binary classifications for a high-level question (detecting entity mentions or syntactic arcs). They find various clustering over different ontologies, such as where BERT has a notion for dates and ELMO has a notion of PERSON tag.

Probing is also criticized in terms of correlation vs. causation. [Vanmassenhove et al., 2017] observes Neural Machine Translations probes and finds mismatches between probes and model's performances. They conclude that part of the information

encoded in sentence vectors might be lost during decoding. This leads to studies for identifying causal relations. Researchers intervene in some parts of representations and observe what is used by the model during both probing and original tasks. [Elazar et al., 2021] also uses interventions and updates model presentations with probe gradients. While this reduces probe performances, they do not observe a decrease in model performance, which leads them to the probes may not indicate the linguistic properties used by the model. On the other hand, [Giulianelli et al., 2018] also updates model representations with probe gradients; while this increases the probe performances, it does not affect the model's original task: language modeling. However, they report that gradient updates from probes increased performance for the number agreement task. Thus they conclude that the model might also use the features identified by probes. [Lasri et al., 2022] also studies what part of features are used by BERT by choosing a task that cannot be solvable for any specific feature. By measuring the task performances, they try to find the representation of that feature by intervening model representation. They study grammatical number usage and find that BERT uses separate encodings for the grammatical number for nouns and verbs.

## *4.2 Morphological Segmentation*

Morphological segmentation, is heavily studied in unsupervised morphology learning. Perhaps the first work [Harris, 1955] is based on *successor frequency* statistics. The idea is: given a list of words; we can analyze each word splitting it into its subwords from the beginning (such as the word *evler* has split into *e,ev,evl,evle,evler*), then we can count how many distinct letters follow the subword among the list of words, and we can define a successor frequency for any subword. The same can be applied from reverse, which will produce predecessor frequency. This will lead to high frequencies for affixes, and in this way, we can discover the morphemes in the language.

Morfessor Baseline [Creutz and Lagus, 2002] and its variants Morfessor FlatCat [Grönroos et al., 2014], Morfessor EM+Prune [Grönroos et al., 2020] are family of

the generative models for learning morphemes of a language. Morfessor Baseline defines a morpheme lexicon for a model using parameters $\theta$. A MAP estimation maximizes the product of the model prior $p(\theta)$ and the data likelihood $p(D|\theta)$ during optimization. Since it assumes independence between morphemes, it relies on unigram language modeling. Its prior follows the Maximum Length Description (MDL) principle and assigns higher probabilities to lexicons with fewer and shorter morphs. It starts with a seed lexicon, including whole words in training data. Then, it splits the word recursively into two segments and selects the split that will result in the smallest total cost. The recursion continues until no split occurs, and leaf nodes become the morphemes that make up the word. As a greedy local search algorithm, it converges to the local optimum. In the Morfessor EM+Prune version, the authors start with a seed lexicon, including one million most frequent subwords in the data, and experiments with different priors. During training, after three sub iterations of EM to estimate the probabilities $p(\theta)$ and $p(D|\theta)$, the lexicon is pruned according to different criteria. The MDL principle has been also studied in different works [Goldsmith, 2001] [Goldsmith, 2006][Lee and Goldsmith, 2016].

Moreover, there are different group of works benefit Adaptor Grammar [Johnson et al., 2006], MorphAGram [Eskander et al., 2020] for unsupervised morphological segmentation. Adaptor Grammars are based on Probabilistic Context Free Grammar (PCFG) framework. PCFG models a grammar $G$ according to predefined symbols and rules, and represents a derivation of a sentence $s$ under $G$ as a parse tree $t$. A sentence $s$ may have multiple derivations, that is, parse trees, and PCFG defines a probability distribution over these parse trees. The probability of a parse tree $t$ is calculated by assigning a probability to each rule in $G$ that expands the parse tree $t$. Adaptor Grammars, grounds on PCFG, take it as base distribution and have an additional "adaptor" function that maps the base distribution to another. This adaptor function helps to adapt the probabilities of a tree in PCFG. For example, [Johnson et al., 2006] uses Pitman-Yor Process [Pitman and Yor, 1997] which makes the posterior probability of the tree proportional to its frequency in the training corpus. Different morphological grammars and predefined symbols are defined to use Adaptor Grammars in unsupervised morphological segmentation. For example

[Sirts and Goldwater, 2013] defines a morphological grammar called compounding grammar that indicates a symbol *word* consists of symbol *Compound*, *Compound* consists of *Prefix Stem Suffix* symbols, where each of them consists of *SubMorph* symbol, and finally *SubMorph* consists of *Char* symbol (Word → Compound, Compound → PrefixStemSuffix, Prefix → SubMorph, Stem → SubMorph, Suffix → SubMorph, Submorph → Char). So after defining the symbols and giving the corpus, the posterior distributions of parse trees are calculated. In [Eskander et al., 2020], the authors explore several morphological grammars for languages and use Pitman-Yor Process as an adaptor function.

### *4.3 Morphological Reinflection*

[Zhou and Neubig, 2017], possibly the most related work to our model, proposes a semi-supervised approach to the task. They design a VAE model with continuous and discrete latent variables, where continuous variables are aimed to represent the lemma, and the discrete variables are aimed to represent the labels of morphological features. For discretization, they use the Gumbel-Softmax trick proposed for adapting sampling from categorical variables in differentiable neural networks by approximating argmax operation with a softmax function based on temperature parameter [Gumbel, 1954][Jang et al., 2016]. A word's encoding vector is classified through multiple multi-layer perceptrons (MLPs), which produces a probability vector over labels of related morphological tag. Then probability vectors are used as weights for the tag embeddings. Then taking tag embeddings as values, they use an attention mechanism in which keys are the concatenation of the previous hidden state of the decoder and mean vector. The tag embeddings are also summed and transformed as an initial hidden state of the decoder and the mean vector. They use supervised data for learning classifiers' parameters; for unsupervised data, they infer the tags. They showed that augmenting supervised data with unsupervised data improves performance. Previous work includes supervised learning techniques. [Durrett and DeNero, 2013] uses alignment and learns edit operations. [Kann and Schütze, 2016] proposes neural approach with encoder-decoder architecture using

soft attention [Bahdanau et al., 2015] with stacked GRUs [Cho et al., 2014]. They feed the encoder with a string containing source tags, source words, and target tags. The decoder is trained to predict the target word form.

The particular case of morphological reinflection is, also called morphological inflection, where the input word is a lemma word, has been also studied by many researchers. [Anastasopoulos and Neubig, 2019] proposes data augmentation, which produces hallucinated data lemma-feature tags-target pairs. In their study, shared substrings longer than three were replaced with random alphabetic characters in a language, leading to hallucinatory triples in lemma-tag form. [Wu et al., 2021] applies transformers [Vaswani et al., 2017] for character-level transduction tasks, which include morphological inflection with type and positional embeddings in the encoder. [Yang et al., 2022] improves that work by applying reverse positional encodings and type embeddings. They also apply the data hallucination technique from [Anastasopoulos and Neubig, 2019] and student-forcing idea from [Nicolai and Silfverberg, 2020] where during training, the model is exposed to model outputs instead of gold labels on some samples to reduce exposure bias.

# Chapter 5

# CONCLUSION & FUTURE WORK

This thesis evaluated morphological learning in four unsupervised NLP models: AE, VAE, CharLM, and VQVAE. We have proposed a VQVAE with continuous and discrete latent variables to represent the lemma in continuous space, whereas the morphological features are quantified using multiple codebooks. The models' performances are evaluated on three tasks: probing, morphological segmentation, and morphological reinflection.

Probing experiments showed that each model's representation outperforms the baseline, indicating that morphological knowledge is encoded in models' representations. Further probings were conducted on continuous and discrete latent variables in our VQVAE model. Our results indicate that morphological features are primarily encoded in the model's codebooks. We, however, did not explore the causal effects of discovered features on the models' actual task performances. Future work may cover these types of experiments to understand whether or not the models are *using* the discovered features for their predictions.

In the morphological segmentation part, we showed that using words' likelihood estimations with a heuristic, generative models VAE and CharLM provide comparable results to baselines and are capable of morpheme identification. However, in our error analysis, we observed that our heuristic misses one-letter morphemes and causes over-segmentation and under-segmentation, degrading the model's performance. We also designed a heuristic that will work for a highly agglutinative language such as Turkish. We further investigated our heuristic for morphological segmentation in Hungarian. We observed a decrease in our performance since the test set includes more one-letter morphemes than the Turkish test set. Future work may cover the improvements of the heuristic and its generalization to other languages.

For morphological reinflection, we applied our VQVAE model in multiple languages: Turkish, Hungarian, Georgian, and Spanish. In order to illustrate the contribution of unsupervised data, we compared the results with the results of the supervised version, and we observed improvements over reinflections for each language. We observed that unsupervised data for Turkish helped the model apply phonological alterations. However, we obtained scores under the baseline for Spanish and Georgian languages. We observed failures not only during reinflection to target words but also during the reconstruction of source words. This may indicate that the model's performance still has room for improvement, for example, by changing the KL annealing process or the capacities of the model's components. We also did our experiments on the datasets where the languages utilize mostly suffixation. Future work may also explore typologically different languages with templatic, non-concatenative morphology, such as Arabic or Navajo. The model needs to be tested whether or not it is biased toward suffixing.

Future work may also cover unsupervised paradigm clustering, unsupervised paradigm completion, and evaluating other unsupervised models.

# BIBLIOGRAPHY

[Adi et al., 2017] Adi, Y., Kermany, E., Belinkov, Y., Lavi, O., and Goldberg, Y. (2017). Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *ArXiv*, abs/1608.04207.

[Ambrose et al., 2010] Ambrose, S. A., Bridges, M. W., Lovett, M. C., DiPietro, M., and Norman, M. K. (2010). How learning works : Seven research-based principles for smart teaching.

[Anastasopoulos and Neubig, 2019] Anastasopoulos, A. and Neubig, G. (2019). Pushing the limits of low-resource morphological inflection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 984–996, Hong Kong, China. Association for Computational Linguistics.

[Bahdanau et al., 2015] Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

[Batsuren et al., 2022] Batsuren, K., Bella, G., Arora, A., Martinovi'c, V., Gorman, K., vZabokrtsk'y, Z., Ganbold, A., vS'arka Dohnalov'a, vSevvc'ikov'a, M., Pelegrinov'a, K., Giunchiglia, F., Cotterell, R., and Vylomova, E. (2022). The sigmorphon 2022 shared task on morpheme segmentation. *ArXiv*, abs/2206.07615.

[Belinkov, 2022] Belinkov, Y. (2022). Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48:207–219.

[Cahill, 2016] Cahill, L. (2016). Computational morphology. In Hippisley, A. and Stump, G., editors, *The Cambridge Handbook of Morphology*, Cambridge Handbooks In Language and Linguistics, pages 820–846. CUP, Cambridge.

[Cho et al., 2014] Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder–decoder approaches. In *SSST@EMNLP*.

[Clark, 2017] Clark, E. V. (2017). Morphology in language acquisition. *The handbook of morphology*, pages 374–389.

[Conneau et al., 2018] Conneau, A., Kruszewski, G., Lample, G., Barrault, L., and Baroni, M. (2018). What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. *ArXiv*, abs/1805.01070.

[Cotterell et al., 2018] Cotterell, R., Kirov, C., Sylak-Glassman, J., Walther, G., Vylomova, E., McCarthy, A. D., Kann, K., Mielke, S. J., Nicolai, G., Silfverberg, M., Yarowsky, D., Eisner, J., and Hulden, M. (2018). The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

[Cotterell et al., 2017] Cotterell, R., Kirov, C., Sylak-Glassman, J., Walther, G., Vylomova, E., Xia, P., Faruqui, M., Kübler, S., Yarowsky, D., Eisner, J., and Hulden, M. (2017). Conll-sigmorphon 2017 shared task: Universal morphological reinflection in 52 languages. In *CoNLL Shared Task*.

[Cotterell et al., 2016a] Cotterell, R., Kirov, C., Sylak-Glassman, J., Yarowsky, D., Eisner, J., and Hulden, M. (2016a). The SIGMORPHON 2016 shared task—morphological reinflection. In *Proceedings of the 2016 Meeting of SIGMORPHON*, Berlin, Germany. Association for Computational Linguistics.

[Cotterell et al., 2016b] Cotterell, R., Kirov, C., Sylak-Glassman, J., Yarowsky, D., Eisner, J., and Hulden, M. (2016b). The sigmorphon 2016 shared task—morphological reinflection. In *SIGMORPHON*.

[Creutz and Lagus, 2002] Creutz, M. and Lagus, K. (2002). Unsupervised discovery of morphemes. *arXiv preprint cs/0205057*.

[Durrett and DeNero, 2013] Durrett, G. and DeNero, J. (2013). Supervised learning of complete morphological paradigms. In *NAACL*.

[Elazar et al., 2021] Elazar, Y., Ravfogel, S., Jacovi, A., and Goldberg, Y. (2021). Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175.

[Eskander et al., 2020] Eskander, R., Callejas, F., Nichols, E., Klavans, J. L., and Muresan, S. (2020). Morphagram, evaluation and framework for unsupervised morphological segmentation. In *LREC*.

[Giulianelli et al., 2018] Giulianelli, M., Harding, J., Mohnert, F., Hupkes, D., and Zuidema, W. H. (2018). Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *BlackboxNLP@EMNLP*.

[Goldsmith, 2001] Goldsmith, J. A. (2001). Linguistica: An automatic morphological analyzer.

[Goldsmith, 2006] Goldsmith, J. A. (2006). An algorithm for the unsupervised learning of morphology. *Natural Language Engineering*, 12:353 – 371.

[Goldsmith et al., 2017] Goldsmith, J. A., Lee, J. L., and Xanthos, A. (2017). Computational learning of morphology.

[Grönroos et al., 2020] Grönroos, S.-A., Virpioja, S., and Kurimo, M. (2020). Morfessor em+ prune: Improved subword segmentation with expectation maximization and pruning. *arXiv preprint arXiv:2003.03131*.

[Grönroos et al., 2014] Grönroos, S.-A., Virpioja, S., Smit, P., and Kurimo, M. (2014). Morfessor flatcat: An hmm-based method for unsupervised and semi-supervised learning of morphology. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1177–1185.

[Gumbel, 1954] Gumbel, E. J. (1954). *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office.

[Harris, 1955] Harris, Z. S. (1955). From phoneme to morpheme. *Language*, 31:32–67.

[Hewitt and Liang, 2019] Hewitt, J. and Liang, P. (2019). Designing and interpreting probes with control tasks. *ArXiv*, abs/1909.03368.

[Hinton and Salakhutdinov, 2006] Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507.

[Hupkes and Zuidema, 2018] Hupkes, D. and Zuidema, W. H. (2018). Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *J. Artif. Intell. Res.*, 61:907–926.

[Jang et al., 2016] Jang, E., Gu, S., and Poole, B. (2016). Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

[Johnson et al., 2006] Johnson, M., Griffiths, T., and Goldwater, S. (2006). Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. *Advances in neural information processing systems*, 19.

[Kann et al., 2020] Kann, K., McCarthy, A. D., Nicolai, G., and Hulden, M. (2020). The sigmorphon 2020 shared task on unsupervised morphological paradigm completion. In *SIGMORPHON*.

[Kann and Schütze, 2016] Kann, K. and Schütze, H. (2016). Med: The lmu system for the sigmorphon 2016 shared task on morphological reinflection. In *SIGMORPHON*.

[Kingma and Welling, 2013] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

[Köhn, 2015] Köhn, A. (2015). What's in an embedding? analyzing word embeddings through multilingual evaluation. In *EMNLP*.

[Koskenniemi, 1983] Koskenniemi, K. (1983). Two-level model for morphological analysis. In *IJCAI*.

[Kurimo et al., 2010] Kurimo, M., Virpioja, S., Turunen, V. T., and Lagus, K. (2010). Morpho challenge competition 2005–2010: evaluations and results. In *ACL 2010*.

[Lasri et al., 2022] Lasri, K., Pimentel, T., Lenci, A., Poibeau, T., and Cotterell, R. (2022). Probing for the usage of grammatical number. In *ACL*.

[Lee and Goldsmith, 2016] Lee, J. and Goldsmith, J. (2016). Linguistica 5: Unsupervised learning of linguistic structure. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 22–26, San Diego, California. Association for Computational Linguistics.

[Mager et al., 2020] Mager, M., cCetinouglu, O., and Kann, K. (2020). Tackling the low-resource challenge for canonical segmentation. In *EMNLP*.

[Michael et al., 2020] Michael, J., Botha, J. A., and Tenney, I. (2020). Asking without telling: Exploring latent ontologies in contextual representations. *ArXiv*, abs/2004.14513.

[Mikolov et al., 2013] Mikolov, T., Chen, K., Corrado, G. S., and Dean, J. (2013). Efficient estimation of word representations in vector space. In *ICLR*.

[Narasimhan et al., 2015] Narasimhan, K., Barzilay, R., and Jaakkola, T. (2015). An unsupervised method for uncovering morphological chains. *Transactions of the Association for Computational Linguistics*, 3:157–167.

[Nicolai and Silfverberg, 2020] Nicolai, G. and Silfverberg, M. (2020). Noise isn't always negative: Countering exposure bias in sequence-to-sequence inflection models. In *COLING*.

[Oflazer, 1993] Oflazer, K. (1993). Two-level description of turkish morphology. *Literary and Linguistic Computing*, 9:137–148.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. (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.

[Pimentel et al., 2021] Pimentel, T., Ryskina, M., Mielke, S. J., Wu, S., Chodroff, E., Leonard, B., Nicolai, G., Ghanggo Ate, Y., Khalifa, S., Habash, N., El-Khaissi, C., Goldman, O., Gasser, M., Lane, W., Coler, M., Oncevay, A., Montoya Samame, J. R., Silva Villegas, G. C., Ek, A., Bernardy, J.-P., Shcherbakov, A., Bayyr-ool, A., Sheifer, K., Ganieva, S., Plugaryov, M., Klyachko, E., Salehi, A., Krizhanovsky, A., Krizhanovsky, N., Vania, C., Ivanova, S., Salchak, A., Straughn, C., Liu, Z., Washington, J. N., Ataman, D., Kieraś, W., Woliński, M., Suhardijanto, T., Stoehr, N., Nuriah, Z., Ratan, S., Tyers, F. M., Ponti, E. M., Aiton, G., Hatcher, R. J., Prud'hommeaux, E., Kumar, R., Hulden, M., Barta, B., Lakatos, D., Szolnok, G., Ács, J., Raj, M., Yarowsky, D., Cotterell, R., Ambridge, B., and Vylomova, E. (2021). Sigmorphon 2021 shared task on morphological reinflection: Generalization across languages. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–259, Online. Association for Computational Linguistics.

[Pimentel et al., 2020] Pimentel, T., Valvoda, J., Maudslay, R. H., Zmigrod, R., Williams, A., and Cotterell, R. (2020). Information-theoretic probing for linguistic structure. *ArXiv*, abs/2004.03061.

[Pitman and Yor, 1997] Pitman, J. and Yor, M. (1997). The two-parameter poisson-

dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, pages 855–900.

[Sahin et al., 2020] Sahin, G. G., Vania, C., Kuznetsov, I., and Gurevych, I. (2020). Multilingual probing tasks for word representations. *Computational Linguistics*, Just Accepted:1–92.

[Shi et al., 2016] Shi, X., Padhi, I., and Knight, K. (2016). Does string-based neural mt learn source syntax? In *EMNLP*.

[Sirts and Goldwater, 2013] Sirts, K. and Goldwater, S. (2013). Minimally-supervised morphological segmentation using adaptor grammars. *Transactions of the Association for Computational Linguistics*, 1:255–266.

[Tenney et al., 2019] Tenney, I., Xia, P., Chen, B., Wang, A., Poliak, A., McCoy, R. T., Kim, N., Durme, B. V., Bowman, S. R., Das, D., and Pavlick, E. (2019). What do you learn from context? probing for sentence structure in contextualized word representations. *ArXiv*, abs/1905.06316.

[Van Den Oord et al., 2017] Van Den Oord, A., Vinyals, O., et al. (2017). Neural discrete representation learning. *Advances in neural information processing systems*, 30.

[Vanmassenhove et al., 2017] Vanmassenhove, E., Du, J., and Way, A. (2017). Investigating 'aspect' in nmt and smt: Translating the english simple past and present perfect.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N. M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *ArXiv*, abs/1706.03762.

[Voita and Titov, 2020] Voita, E. and Titov, I. (2020). Information-theoretic probing with minimum description length. *ArXiv*, abs/2003.12298.

[Wiemerslage et al., 2021] Wiemerslage, A., McCarthy, A. D., Erdmann, A., Nicolai, G., Agirrezabal, M., Silfverberg, M., Hulden, M., and Kann, K. (2021). Findings of the sigmorphon 2021 shared task on unsupervised morphological paradigm clustering. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 72–81.

[Wu et al., 2021] Wu, S., Cotterell, R., and Hulden, M. (2021). Applying the transformer to character-level transduction. In *EACL*.

[Yang et al., 2022] Yang, C., Yang, R. R., Nicolai, G., and Silfverberg, M. (2022). Generalizing morphological inflection systems to unseen lemmas. *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*.

[Zhou and Neubig, 2017] Zhou, C. and Neubig, G. (2017). Multi-space variational encoder-decoders for semi-supervised labeled sequence transduction. *arXiv preprint arXiv:1704.01691*.

# Appendix A

We also use our morphological segmentation heuristic in Hungarian, described in chapter 3.2. To do this, we use a Hungarian dataset from SIGMORPHON 2022 Shared Task Data on Canonical Segmentation. We filter the words where the canonical and surface segments are the same; that is, the word's orthography is formed from its canonical segments. We train VAE and CharLM with the given settings in chapter 3.2. We train the Morphagram model with standard grammar as PrefixStemSuffix + Submorphs (PrStSu+SM).

We observe that, for both VAE and CharLM, our gap between the baseline model Morphagram increases because the Hungarian test set includes more one-letter morphemes, and our heuristics can not identify them besides again suffering from over-segmentation and under-segmentation. However, our model may still be a good candidate with improvements to the heuristic.

| **Language** | **Train (# of words)** | **Test (# of words)** |
|---|---|---|
| Hungarian | 61205 | 1000 |

Table A.1: Data statistics for morphological segmentation in Hungarian

| **Model** | **Precision** | **Recall** | **F1** |
|---|---|---|---|
| *CharLM | 59.55% | 58.02% | 58.78% |
| *VAE | 58.90% | 57.93% | 58.41% |
| Morphagram | 80.79% | 67.09% | **73.31**% |

Table A.2: Morphological segmentation results on Hungarian test set. *:Our models