# Automatic Grammatical Case Prediction for Template Filling in Case-Marking Languages: Implementation and Evaluation for Finnish

**Johannes Laurmaa**
johlaurmaa@gmail.com

## Abstract

Automatically generating grammatically correct sentences in case-marking languages is hard because nominal case inflection depends on context. In template-based generation, placeholders must be inflected to the *right case* before insertion, otherwise the result is ungrammatical. We formalise this *case selection* problem for template slots and present a practical, data-driven solution designed for morphologically rich, case-marking languages, and apply it to Finnish. We automatically derive training instances from raw text via morphological analysis, and fine-tune transformer encoders to predict a distribution over 14 grammatical cases, with and without lemma conditioning. The predicted case is then realized by a morphological generator at deployment. On a held-out test set in the lemma-conditioned setting, our model attains 89.1% precision, 81.1% recall, and 84.2% F1, with recall@3 of 93.3% (macro averages). The probability outputs support abstention and top-$k$ suggestion User Interfaces, enabling robust, lightweight template filling for production use in multiple domains, such as customer messaging. The pipeline assumes only access to raw text plus a morphological analyzer and generator, and can be applied to other languages with productive case systems.

## 1 Introduction

Generating natural-sounding text in highly inflected languages poses challenges that are often overlooked in languages like English. For example, Finnish relies on a rich system of grammatical cases, so that a word like `Helsinki` takes different endings depending on its role in the sentence. In English, an email template such as

```
Your trip to [CITY] is starting
```

allows any city name to be dropped in directly—no changes needed. But in Finnish, you can't simply insert `Helsinki` into the same sentence structure, because the word ending will depend on the grammatical case. The correct form is

```
Matkasi Helsinkiin alkaa
```

, where `Helsinki` has been inflected to `Helsinkiin`, the illative case, to indicate movement towards the city.

While it is technically possible for template authors to hand-select the case for each placeholder (slow and error-prone), in practice templates are often rewritten to keep the placeholder nominative—at the cost of fluency and ongoing content maintenance (e.g. reworded as `Matkasi kohteeseen Helsinki alkaa`[1], which would be translated as `Your trip to the destination Helsinki is starting`).

This is a simple example, but the problem is widespread in template-based text generation, which is common in applications ranging from automatic email messages to travel apps and customer notifications.

These workarounds are inefficient and underscore the need for automated solutions for inflected languages that can select the correct word form in context.

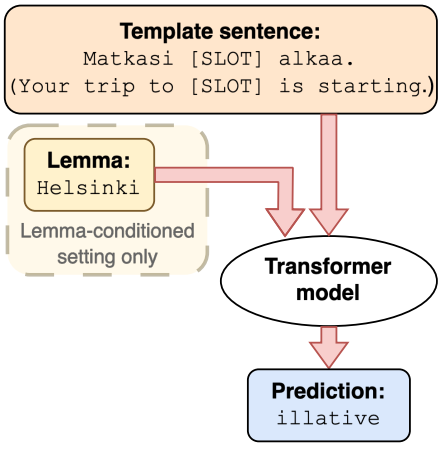


Figure 1: Our approach of grammatical case prediction.

[1] Real-world example of an email



*Proceedings of the 8th Workshop on Research in Computational Linguistic Typology and Multilingual NLP*, pages 1–11

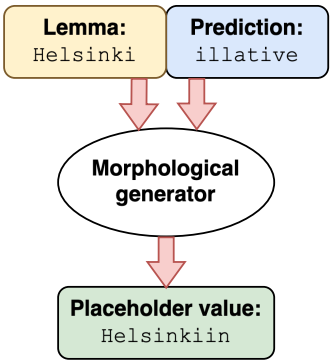


Figure 2: Surface realisation left to the morphological generator.

In this paper, we present a machine learning approach for predicting the required grammatical case when filling placeholders in Finnish sentence templates. Our approach creates training data from any Finnish text and uses transformer-based models to predict the correct grammatical case for incomplete template sentences (cf. Figure 1). We publish a high-performing model for this task and discuss both the challenges and the remaining ambiguities involved. Our goal is a deployable solution: predict the correct case with confidence scores, expose top-$k$ when needed, and offload surface realisation to a morphological generator (cf. Figure 2).

**Contributions.**

- Formalize Finnish *case selection* with slot-only and lemma-conditioned settings.
- Automatic *dataset construction* from raw Finnish text via morphological analysis.
- Trained transformer encoder model predicting grammatical case probabilities.

## 2 Related work

### 2.1 Template-based NLG and morphology

Template-based Natural Language Generation (NLG) typically relies on sentence skeletons with slots filled at runtime (Van Deemter et al., 2005). For morphologically rich languages, template systems need to offload morphology to separate components that realise surface forms given a lemma and the sentence context. Dušek and Jurcícek (2019) already suggest approaches to this problem: using language models to select correct inflected forms, and using sequence-to-sequence models generating sequences of lemmas and morphological tags before passing them to a morphological generator. We adapt a similar approach to the Finnish language, but using language models to select correct grammatical cases and leaving the surface realisation to a morphological generator, also touching on the concept of aleatoric uncertainty and comparing prediction accuracy with and without lemma conditioning.

### 2.2 Finnish morphology: analysis and generation

Two-level finite-state morphology and HFST-based tools provide one established way to model Finnish morphology (Hämäläinen and Alnajjar, 2021). Omorfi provides a finite-state lexicon and analyzer for Finnish, supporting both morphological analysis and generation (Pirinen, 2015). UralicNLP exposes these analyzers and generators via a Python API, returning lemmas and rich feature bundles (including case) and generating inflected forms from lemma+features (Hämäläinen, 2019). Morphological generators solve the *realisation* problem: given lemma and features, output the correct surface form. They do not decide which case to use in a particular sentence, which is the task we address in this paper.

### 2.3 Finnish transformer models and LLMs

Virtanen et al. (2019) introduced FinBERT, a BERT-style model trained on large Finnish corpora, which outperforms multilingual BERT on core Finnish NLP tasks. We build directly on this line by fine-tuning FinBERT as a case classifier and comparing it to a multilingual model (XLM-R).

### 2.4 Positioning

Our setup sits between morphological tagging and morphological generation. In tagging, the model sees fully inflected tokens and assigns case labels to them in context. Here, the surface form is absent: we see a sentence with a marked slot, optionally with a lemma, and predict the case that should be used for that slot.

Finnish morphological generators, in turn, take lemma+features (including case) as input and realise the surface form. Our method is intended as an upstream component: it decides which case to request from the generator for a given slot in context.

To our knowledge, automatic case selection for template slots has not been studied directly. This work applies the idea to Finnish and integrates it with existing morphology tools in a template-filling setting.

| Case | Example | Rough meaning |
|---|---|---|
| Core cases | | |
| Nominative | `talo` | house |
| Genitive | `talon` | of the house |
| Partitive | `taloa` | (some) house |
| Accusative | `talo(n)` | the house (object) |
| Internal locative cases | | |
| Inessive | `talossa` | in the house |
| Elative | `talosta` | out of the house |
| Illative | `taloon` | into the house |
| External locative cases | | |
| Adessive | `talolla` | at the house |
| Ablative | `talolta` | from the house |
| Allative | `talolle` | to the house |
| Other cases | | |
| Essive | `talona` | as a house |
| Translative | `taloksi` | into a house |
| Abessive | `talotta` | without a house |
| Instructive | `taloin` | by means of houses |
| Comitative | `taloineen` | with houses |

Table 1: Finnish grammatical cases with the word `talo` (house) as an example.

## 3 Methodology

### 3.1 Grammatical cases

It is generally agreed that the Finnish language has 15 grammatical cases. Table 1 shows the grammatical cases in Finnish with the singular form of the word `talo` (house) used as an example.

### 3.2 Problem formulation

We cast Finnish template filling as a *case selection* task. The input is a sentence template $s$ containing an annotated *slot* for a head noun (a noun in its baseform). The model outputs a categorical distribution over grammatical cases for that slot. In training, we expose both renderings (slot-only and lemma-conditioned) so the model learns to operate under either input condition.

**Scope (what we predict).** We predict a case label $y \in \mathcal{Y}$ for the head noun. The label set consists of 14 cases:

```
Nom, Gen, Par, Ine, Ela,
Ill, Ade, Abl, All, Ess,
Tra, Abe, Ins, Com
```

We left out the accusative class, as in contemporary Finnish, object marking is often realized as nominative or genitive. Surface-form realisation (inflecting a lemma into the final form of the word) is delegated to a morphological generator at deployment time. We also do not predict number, possessive suffixes or clitics, as these can also be handled by the morphological generator.

**Supervision (single label).** Training and test instances carry a *single* case $y^\star$ derived from morphological analysis of observed text. Although multiple cases can be pragmatically plausible for the same context (aleatoric uncertainty), supervision remains single-label.

**Inference settings.** We consider two settings that differ in what is provided about the slot content:

#### 3.2.1 Slot-only (no-lemma) setting

Only the template with a slot marker is given; the model must infer the case from context alone.

> **Input:** Haluatko muuttaa [SLOT]?
> **Translation:** *Do you want to move [SLOT]?*
> **Label:** `Ill`

#### 3.2.2 Lemma-conditioned setting

The lemma $\ell$ of the noun to be inserted (e.g., a city name) is also provided, which can inform case preferences.

> **Input:** Haluatko muuttaa [SLOT: Helsinki]?
> **Translation:** *Do you want to move [SLOT: Helsinki]?*
> **Label:** `Ill`

**Output and evaluation.** The model estimates $p(y \mid s)$ or $p(y \mid s, \ell)$. We report top-1 accuracy (via $\arg\max_y p(y \mid \cdot)$) and top-$k$ accuracy. For applications, systems may surface top-$k$ candidates or abstain under low confidence.

### 3.3 Word inflection

Once the correct case has been predicted, the word can be inflected to the selected case using morphological generation tools (Hämäläinen, 2019; Alnajjar and Hämäläinen, 2023), cf. Figure 2.

### 3.4 Inherent uncertainty

Note that the setups above do not always admit a unique solution, as the correct grammatical case will depend on the intent of the writer. For the

example shown above, there are other possible solutions besides the illative case:

> **Label:** `Ill`
> **Output:** Haluatko muuttaa Helsinkiin?
> **Translation:** *Do you want to move to Helsinki?*

> **Label:** `Ela`
> **Output:** Haluatko muuttaa Helsingistä?
> **Translation:** *Do you want to move out of Helsinki?*

> **Label:** `Par`
> **Output:** Haluatko muuttaa Helsinkiä?
> **Translation:** *Do you want to change Helsinki?*

The multiplicity of solutions must be taken into account in application design. The model must be able to predict multiple classes with varying probability. This is a probabilistic single-label classification task with aleatoric uncertainty, so the expected optimal output is a probability distribution over classes rather than a single hard prediction.

While it cannot fully clear the uncertainty, the lemma-conditioned formulation will help alleviate it by bringing extra information about the user's intent. In the example above, knowing that the placeholder will contain a city name makes locative cases (elative / illative) more likely.

### 3.5 Dataset creation

We outline a simple, corpus-agnostic recipe to build supervision for case selection:

1. **Collect Finnish text.** Any raw Finnish text from the target domain is suitable.
2. **Run morphological analysis.** Using a Finnish morphological analyser, obtain for each token, its lemma (baseform) and grammatical case (when applicable), along with sentence boundaries.
3. **Construct instances.** For randomly selected target nouns in each sentence, create inputs in two views:
   - *Slot-only*: replace the surface token with a special marker (e.g., `Helsinkiin` replaced by `[MASK]`).
   - *Lemma-conditioned*: replace the surface token with its lemma (e.g., `Helsinkiin` replaced by `Helsinki`).

   The target label is the noun's grammatical case extracted from the morphological analysis; supervision is single-label.

Concrete choices (corpus, splits, and any training specifics) are detailed in Section 4.1.

### 3.6 Model selection

We considered:

- The problem highly depends on the context, relations among words and writer intent.
- The task exhibits aleatoric ambiguity: even with full context, multiple labels may be plausible. The model should therefore output a categorical distribution $p(y|x)$ rather than a hard label, and expose top-k classes with their probabilities.
- We aim for a lightweight model that can be easily deployed in production.

Given these specifications, we opted to use a Transformer encoder trained on Finnish or multilingual data (e.g., FinBERT or XLM-R) fine-tuned for K-way single-label classification with a softmax head. Top-k probabilities are returned at inference, with an optional post-hoc probability calibration.

## 4 Experiments

### 4.1 Experimental setup

As source data, we use Finnish news articles from Yleisradio (Yle) released via Kielipankki (Yleisradio, 2022). We utilise the sentence boundaries and morphological annotations already provided within the corpus's VRT format.

**Split by year to prevent leakage.** We train and validate on the **2019** portion of the corpus and evaluate on the **2021** portion only. This ensures no document- or sentence-level overlap across train and test. The 2019 slice contains **2,183,722** sentences; the 2021 slice contains **2,110,654** sentences.

During training, a held-out subset of 2019 is reserved for validation (used for early stopping and model selection). We used a max sequence length of 128 tokens with truncation and padding, a cross-entropy objective, AdamW as optimizer, trained

on 1 epoch with a batch size of 16, a learning rate of $2 \cdot 10^{-5}$ and weight decay 0.01. We marked template placeholders in the slot-only setting with the '[MASK]' token.

#### 4.1.1 Preprocessing and instance sampling

We convert running text into classification instances as follows:

**Candidate head nouns.** We run an external morphological analyser over each sentence and *preselect only nouns* as candidate heads. Tokens analyzed as accusative objects are *excluded* (we do not model accusative; cf. *Problem formulation* section).

**Class-aware sampling.** To control class imbalance in downstream training, each noun token is assigned a probability of being selected as the supervised slot. The proportion of samples in each class in the evaluation set is shown in Table 6. By default, we set nouns to have a **20%** probability to be selected as prediction sample. For rare cases, we up-weight selection to guarantee sufficient coverage: **100%** for *comitative* and *abessive*, and **30%** for *instructive*. This concentrates supervision on long-tail labels without altering the target label distribution for common cases and avoids severely undertrained heads for rare labels.

**Slot rendering (two settings).** For each selected noun we create *two* training views:

- **slot-only** by replacing the token with a special marker `[MASK]` and
- **lemma-conditioned** by replacing the surface form with its *lemma*.

We chose to sample these views with a **50/50** proportion so the model learns both settings jointly.

**Target labels.** The target case label for the slot is taken directly from the morphological analysis provided with the Yle corpus. We treat the corpus analysis as the reference and *do not* attempt to resolve alternative parses; supervision is single-label as mentioned in section 3.4.

**Baselines.** As baselines, we include two lightweight heuristics. First, we use a pure prior baseline that always outputs the global class probability found in the training data, close to the one mentioned in Table 6. Second, we use an adposition-based rule baseline. In Finnish, as in several other languages, the case of a word can be influenced by a pre- or postposition. For example, the postposition *lähellä* (*close to* in English) generally follows a word inflected to the genitive case (e.g. *talon lähellä*, meaning *close to the house*). We collected a list of adpositions and their governing cases in *Iso suomen kielioppi* (§692-720) (Hakulinen, 2004), see Table 7. At inference, this baseline scans for an adposition adjacent to the slot and predicts the governed case using the appropriate pre/post rule; if no governing adposition is present, we default to the prior baseline.

Our baselines are slot-only by design, focusing on identifying surface-level markers from the adjacent context. Incorporating the lemma into these baselines would require complex rule-based logic to map specific nouns to their most probable cases. The low performance of these baselines reflects the task's inherent complexity, which demands models capable of capturing deeper linguistic dependencies.

## 5 Results

We evaluate two encoder models (FinBERT, XLM-R) under two settings: (i) *No-lemma* (slot-only) and (ii) *Lemma-conditioned*. Unless noted, metrics are computed over the 14-case label set (no separate accusative).

### 5.1 Overall performance

All results use the instance construction procedure described in Sections 3.5 and 4.1.

Table 2 summarizes main metrics. FinBERT in the lemma-conditioned setting attains the best accuracy and F1. Providing the lemma yields a substantial gain over the slot-only setting.

### 5.2 Per-class and confusion analysis

Tables 3 and 4 show confusion matrices for the FinBERT model in the slot-only and lemma-conditioned settings, respectively. The dominant confusions cluster within the locative system: most confused classes are adessive and inessive, particularly in the slot-only prediction (see example predictions in table 8 in the appendix). This is understandable, as the Finnish language treats place names either as something you go "into" (internal locative cases, e.g. `Helsinki` → `Helsinkiin`) or "at" (external locative cases, e.g.

| | Lemma-conditioned | | Slot-only | | Baselines | |
|---|---|---|---|---|---|---|
| Metric | FinBERT | XLM-R | FinBERT | XLM-R | Majority class | Adpositions |
| **Top-1** | | | | | | |
| Precision | 89.1% | 85.2% | 73.3% | 51.9% | 2.5% | 19.4% |
| Recall | 81.1% | 72.2% | 63.0% | 41.6% | 7.1% | 7.7% |
| F1 score | 84.2% | 75.3% | 66.7% | 44.7% | 3.7% | 5.0% |
| Accuracy | 91.4% | 87.8% | 82.6% | 80.8% | 34.9% | 35.7% |
| **Top-3** | | | | | | |
| Recall | 93.3% | 90.8% | 84.4% | 73.9% | 21.4% | 21.7% |
| Accuracy | 96.0% | 94.1% | 90.0% | 89.6% | 72.4% | 72.6% |

Table 2: Model performance in lemma-conditioned and slot-only settings. Baselines are slot-only heuristics. Precision, recall, and F1 are macro-averaged over the 14 classes.

| Actual \ Predicted | Abe | Abl | Ade | All | Com | Ela | Ess | Gen | Ill | Ine | Ins | Nom | Par | Tra |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Abe | 38 | 0 | 3 | 3 | 0 | 2 | 1 | 8 | 9 | 13 | 0 | 7 | 14 | 1 |
| Abl | 0 | 48 | 4 | 2 | 0 | 17 | 2 | 6 | 3 | 9 | 1 | 4 | 3 | 0 |
| Ade | 0 | 0 | 45 | 1 | 0 | 2 | 5 | 6 | 2 | 31 | 0 | 4 | 3 | 0 |
| All | 0 | 0 | 2 | 56 | 0 | 3 | 1 | 6 | 19 | 6 | 0 | 3 | 2 | 5 |
| Com | 0 | 1 | 4 | 1 | 26 | 2 | 2 | 14 | 2 | 19 | 1 | 23 | 3 | 0 |
| Ela | 0 | 2 | 1 | 1 | 0 | 77 | 1 | 5 | 2 | 3 | 0 | 4 | 4 | 0 |
| Ess | 0 | 0 | 3 | 1 | 0 | 2 | 62 | 6 | 1 | 15 | 0 | 7 | 2 | 0 |
| Gen | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 91 | 1 | 1 | 0 | 4 | 2 | 0 |
| Ill | 0 | 0 | 1 | 3 | 0 | 2 | 1 | 5 | 80 | 3 | 0 | 2 | 2 | 1 |
| Ine | 0 | 1 | 5 | 1 | 0 | 3 | 3 | 7 | 2 | 69 | 0 | 6 | 3 | 0 |
| Ins | 0 | 1 | 5 | 1 | 0 | 2 | 2 | 8 | 3 | 7 | 63 | 6 | 2 | 0 |
| Nom | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 1 | 0 | 92 | 2 | 0 |
| Par | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 3 | 1 | 1 | 0 | 7 | 86 | 0 |
| Tra | 1 | 1 | 2 | 2 | 0 | 4 | 3 | 7 | 17 | 6 | 0 | 4 | 4 | 50 |

Table 3: Confusion matrix for FinBERT in slot-only setting

`Tampere` → `Tampereelle`). This pattern also accounts for the confusion of ablative with elative, and illative with allative. Since the choice often depends on the lemma itself, this confusion is significantly alleviated in the lemma-conditioned setting, where the model is able to use the lemma for disambiguation.

Beyond locatives, the model often misclassifies long-tail cases (e.g., abessive, comitative, translative) as their higher-frequency counterparts (e.g., inessive, genitive, nominative). These errors likely stem from shared syntactic roles; for instance, the translative and illative both frequently denote a terminal state or destination in a sentence. In such cases, the model defaults to the most statistically probable category when the specific semantic marker of the rare case is absent from the context.

### 5.3 Top-3 performance

Table 5 shows the top-3 performance metrics by case in the lemma-conditioned setting. Top-3 performance could be relevant for applications where the user can select from a couple of options from a dropdown menu, for example. The model achieves excellent recall@3 values of 93.3% (macro average) and 96.0% (micro average).

## 6 Discussion

### 6.1 What about LLMs?

In recent years, LLMs have become increasingly popular and they allow far greater flexibility in generating text. However, flexibility is not always

| Actual \ Predicted | Abe | Abl | Ade | All | Com | Ela | Ess | Gen | Ill | Ine | Ins | Nom | Par | Tra |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Abe | 81 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 7 | 7 | 0 |
| Abl | 0 | 78 | 4 | 2 | 0 | 5 | 1 | 3 | 1 | 2 | 0 | 3 | 0 | 0 |
| Ade | 0 | 1 | 85 | 0 | 0 | 1 | 0 | 3 | 1 | 5 | 0 | 3 | 1 | 0 |
| All | 0 | 1 | 3 | 80 | 0 | 1 | 1 | 3 | 6 | 2 | 0 | 3 | 1 | 0 |
| Com | 0 | 0 | 2 | 2 | 35 | 5 | 4 | 9 | 1 | 3 | 0 | 39 | 2 | 0 |
| Ela | 0 | 1 | 0 | 0 | 0 | 85 | 0 | 2 | 1 | 2 | 0 | 4 | 3 | 0 |
| Ess | 0 | 0 | 0 | 0 | 0 | 1 | 84 | 2 | 1 | 1 | 0 | 9 | 1 | 0 |
| Gen | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 94 | 0 | 1 | 0 | 3 | 1 | 0 |
| Ill | 0 | 0 | 1 | 2 | 0 | 1 | 1 | 2 | 86 | 3 | 0 | 3 | 2 | 0 |
| Ine | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 3 | 2 | 86 | 0 | 4 | 1 | 0 |
| Ins | 0 | 0 | 5 | 0 | 0 | 0 | 1 | 5 | 2 | 1 | 79 | 6 | 0 | 0 |
| Nom | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 97 | 1 | 0 |
| Par | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 8 | 88 | 0 |
| Tra | 0 | 0 | 1 | 1 | 0 | 2 | 4 | 2 | 5 | 1 | 0 | 5 | 1 | 77 |

Table 4: Confusion matrix for FinBERT in lemma-conditioned setting

Table 5: Top-3 performance metrics by case (lemma-conditioned setting)

| Case | Recall @ 3 |
|---|---|
| Abe | 97% |
| Abl | 95% |
| Ade | 93% |
| All | 94% |
| Com | 71% |
| Ela | 95% |
| Ess | 94% |
| Gen | 96% |
| Ill | 95% |
| Ine | 93% |
| Ins | 95% |
| Nom | 98% |
| Par | 96% |
| Tra | 97% |
| Micro avg | 96% |
| Macro avg | 93% |

Table 6: Number of examples per class in the evaluation dataset

| Case | Number of examples | Percentage |
|---|---|---|
| Nom | 75305 | 33.4% |
| Gen | 54045 | 23.9% |
| Par | 31256 | 13.9% |
| Ine | 17407 | 7.7% |
| Ill | 11045 | 4.9% |
| Ela | 10907 | 4.8% |
| Ade | 10284 | 4.6% |
| Ess | 4910 | 2.2% |
| All | 4885 | 2.2% |
| Abl | 2132 | 0.9% |
| Tra | 1498 | 0.7% |
| Ins | 1203 | 0.5% |
| Abe | 289 | 0.1% |
| Com | 230 | 0.1% |
| Total | 225396 | 100.0% |

desired in all applications. Template filling is still a simple and effective method when more control is needed over what is being sent to the recipient.

LLMs could also be used for inflection of placeholder values in templates. They are also easily accessible and usable by anyone through APIs. However, we argue that this would not be as lightweight as our approach. Another drawback is that off-the-shelf commercial LLMs often only predict the top-1 result. They would be less flexible in producing probabilities of multiple possible grammatical cases, as we do in our approach.

We still performed an experiment comparing our approach with the performance of a state-of-the-art LLM on this task. Details of the experiment are in Appendix A.

Table 7: Adpositions and their case government, used as our baseline (from §692-710, *Iso suomen kielioppi* (Hakulinen, 2004))

| § | Adpositions | Neighbour case(s) | Position(s) |
|---|---|---|---|
| 692 | luona, luota, asemesta, takia, vuoksi | genitive | postposition |
| | varten | partitive | postposition |
| | vastoin | genitive | preposition |
| | vasten | partitive | preposition or postposition |
| 696 | lukuun ottamatta | partitive | preposition or postposition |
| | huolimatta, riippumatta | elative | preposition or postposition |
| | katsomatta | illative | postposition |
| 697 | mennessä | illative<br>allative | postposition |
| | kuluessa, kuluttua | genitive | postposition |
| | verrattuna, suhteutettuna | illative | preposition or postposition |
| 698 | alkaen, lähtien | elative<br>ablative | preposition or postposition |
| | lukien, laskien, pitäen | elative | postposition |
| | riippuen, johtuen | elative | preposition or postposition |
| | katsoen, nähden, perustuen, liittyen | illative | preposition or postposition |
| | koskien | partitive | preposition or postposition |
| | mukaan lukien, huomioon ottaen, pois lukien | nominative | preposition or postposition |
| 702 | halki, poikki, läpi, ali, yli, alla, alle, alta, yllä, ylle, yltä | genitive | preposition or postposition |
| 703 | lähellä, edellä, vastapäätä | partitive<br>genitive | preposition or postposition<br>postposition |
| | ympäri | partitive<br>genitive | preposition or postposition |
| 704 | keskellä, keskelle, keskeltä | partitive<br>genitive | preposition<br>postposition |
| | kesken | partitive<br>genitive | preposition or postposition |
| 705 | pitkin, kohti, kohden, vastaan, vailla, vaille | partitive | preposition or postposition |
| 710 | päin | partitive | preposition or postposition |
| 708–709 | paitsi | partitive | preposition or postposition |
| | kautta | genitive | preposition or postposition |
| 693 | suhteessa | illative | preposition |

## 7 Conclusion

In this paper we framed template filling in morphologically rich languages as a grammatical case selection task for a head-noun slots and proposed a practical encoder-based solution. Instead of handwritten inflection rules, our system predicts a probability distribution over 14 grammatical cases from raw text context and delegates surface realisation to an external morphological generator. The supervision pipeline is corpus-agnostic: it harvests training instances from morphologically analyzed texts with minimal manual effort.

On held-out Finnish news data, a fine-tuned FinBERT model in the lemma-conditioned setting achieves 89.1% precision, 81.1% recall, 84.2% macro F1 and 91.4% accuracy, with macro recall@3 of 93.3%. These results show that accurate case selection is feasible with a lightweight model, and that providing the lemma substantially reduces confusions within the Finnish locative system. The strong top-3 performance makes the approach particularly suitable for interfaces that can present a small set of alternatives or abstain under low confidence.

A small comparison with a modern LLM indicates that a specialised encoder is competitive on this focused task. Overall, the pipeline we describe—automatic extraction of case-labelled slots,

lemma-aware slot rendering, and probabilistic case prediction—provides a reusable recipe for Finnish, but also for other morphologically rich languages such as Estonian, Hungarian, or Czech. By decoupling high-level case selection from surface realization, we offer a robust pathway for improving grammaticality in typologically diverse languages with comparable resources.

## Limitations

While we proposed a practical solution for grammatical case detection for Finnish templates, several limitations remain.

Our formulation as a single-label prediction task inherits inherent aleatoric uncertainty. Often, several predictions are plausible depending on the writer's intent. In this work, we do not estimate an upper bound on achievable performance under such ambiguity, so it remains unclear how close the reported scores are to the best possible performance without access to user intent.

Second, our experiments are run on news texts. The grammatical case distribution and typical syntactic patterns learned by our model may differ from those typical in the application domain (e.g. in marketing emails or app notifications). We do not quantify how much performance would degrade or shift when moving from news to such domains.

Third, our study is currently restricted to Finnish, and we do not present cross-lingual experiments. Our pipeline could be run on other languages where nouns are regularly inflected to different grammatical cases depending on context (e.g. other Uralic languages, Czech). The effectiveness of the approach in these languages remains to be demonstrated.

Finally, we treat each slot independently, predicting a case for a single head noun at a time. In real templates, multiple placeholders may appear in the same sentence, and the preferred grammatical case for one slot can depend on the choice made for another. Our current model does not enforce global consistency: selecting top-1 predictions for different slots may not jointly produce a coherent sentence.

## References


Khalid Alnajjar and Mika Hämäläinen. 2023. Pyhfst: A pure python implementation of hfst. pages 32–35.

Ondřej Dušek and Filip Jurcícek. 2019. Neural generation for czech: Data and baselines. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 563–574.

Auli Hakulinen. 2004. Iso suomen kielioppi.

Mika Hämäläinen and Khalid Alnajjar. 2021. The current state of finnish nlp. *arXiv preprint arXiv:2109.11326*.

Mika Hämäläinen. 2019. UralicNLP: An NLP library for Uralic languages. *Journal of Open Source Software*, 4(37):1345.

Tommi A Pirinen. 2015. Development and use of computational morphology of finnish in the open source and open science era: Notes on experiences with omorfi development. *Finnish Journal of Linguistics*, (28):381–393.

Kees Van Deemter, Emiel Krahmer, and Mariët Theune. 2005. Squibs and discussions: Real versus template-based natural language generation: A false opposition? *Computational linguistics*, 31(1):15–24.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: Bert for finnish. *arXiv preprint arXiv:1912.07076*.

Yleisradio. 2022. Ylen suomenkielinen uutisarkisto 2019-2021, Korp.

| Input sentence | Input lemma | Predicted case | Correct? |
|---|---|---|---|
| Posti sulkee ovensa Turun [SLOT]<br>*(The post office closes its doors Turku [SLOT])* | N/A | Inessive (-ssa) | ✗ |
| | Eerikinkatu<br>*(Eric Street)* | Adessive (-lla) | ✓ |
| Koronavirus on koetellut Etelä-Afrikkaa eniten Afrikan [SLOT]<br>*(Coronavirus has hit South Africa the hardest Africa [SLOT])* | N/A | Inessive (-ssa) | ✗ |
| | mantere<br>*(continent)* | Adessive (-lla) | ✓ |
| Ensimmäiset merkit taudista havaittiin [SLOT]<br>*(The first signs of the illness were detected [SLOT])* | N/A | Inessive (-ssa) | ✗ |
| | viikonloppu<br>*(weekend)* | Essive (-na) | ✓ |
| Työttömien mielenosoitus laajeni [SLOT] 1990-luvulla<br>*(The unemployed worker's protest spread [SLOT] in the 1990s)* | N/A | Illative (-een) | ✗ |
| | Mannerheimintie<br>*(Mannerheim Street)* | Allative (-lle) | ✓ |
| Kläbo hiihti [SLOT] ja joukkuetoveri tuli toisena maaliin<br>*(Kläbo skied [SLOT] and his teammate finished second)* | N/A | Illative (-een) | ✗ |
| | ykkönen<br>*(first place)* | Translative (-ksi) | ✓ |

Table 8: Examples of predictions from the most confused predicted classes in the slot-only setting. In these examples, all the predicted cases can be grammatically correct, however only the lemma-conditioned predictions match the groundtruth.

| Prompt | Precision | Recall | F1 | Accuracy |
|---|---|---|---|---|
| #1 | 48.2% | 35.8% | 34.9% | 35.8% |
| #2 | 77.1% | 71.6% | 71.3% | 71.9% |
| #3 | 81.1% | 76.6% | 76.7% | 77.3% |

Table 9: GPT-5 performance (macro averages) on the lemma-conditioned task.

## A LLM Experiment Details

We evaluated OpenAI's GPT-5 model on the lemma-conditioned task using three different prompting strategies (Figures 3, 4 and 5). We sampled 1400 instances from the evaluation set, with 100 examples per case. We ran predictions using the `gpt-5-2025-08-07` model with default parameters. For prompts where the model outputs an inflected word or a full sentence, we use Uralic-NLP's morphological analyzer (Hämäläinen, 2019) to recover the predicted grammatical case.

Table 9 reports macro-averaged top-1 performance. The LLM performs worst when asked to predict the case label directly (Prompt 1), better when asked to output only the inflected word (Prompt 2), and best when rewriting the whole sentence with the inflected form (Prompt 3). Even in the strongest setting, GPT-5 remains slightly below our FinBERT lemma-conditioned model.

These results suggest that a heavier LLM does not automatically outperform a specialised encoder on this focused case selection task. The encoder models remain an attractive option: they are relatively lightweight, achieve higher accuracy on our data, and naturally provide a full probability distribution over cases. In contrast, most off-the-shelf LLM APIs expose only top-1 outputs and do not directly return class probabilities.

```
Your task is to detect the matching
    grammatical case in Finnish
    sentences. You will be given a
    sentence with a placeholder. Your
    output should be grammatical case
    that best fits words that are
    inserted in the placeholder.
The possible cases are Nom, Gen, Par,
    Ine, Ela, Ill, Ade, Abl, All, Ess,
    Tra, Abe, Ins, Com.

# Example

Input: `Soitin eilen [kaveri].`
Output: `All`

# Task
Input: `{input_sentence}`
```

Figure 3: Prompt 1: Predict grammatical case name directly.

```
You will be given a sentence in Finnish.
     Your task is to inflect the word in
     the brackets to the correct
    grammatical case. Your output should
     contain the inflected word.

# Example
Input: `Soitin eilen [kaveri].`
Output: `kaverille`

# Task
Input: `{input_sentence}`
```

Figure 4: Prompt 2: Inflect word only.

```
You will be given a sentence in Finnish.
     Your task is to inflect the word in
     the brackets to the correct
    grammatical case. Your output should
     contain the corrected sentence.

# Example
Input: `Soitin eilen [kaveri].`
Output: `Soitin eilen kaverille.`

# Task
Input: `{input_sentence}`
```

Figure 5: Prompt 3: Rewrite whole sentence with word inflected.