Multilingual Interlinear Glossing in the Grammatical Framework

Adrian De Lon[1,2] **Inari Listenmaa**[3]
[1]AI4REASON, Prague, Czechia
[2]Logic Group, University of Bonn, Bonn, Germany
[3]Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden
adelon@uni-bonn.de inari@chalmers.se

Abstract

We present an approach to automated glossing that parses sentences into abstract trees and generates parallel linearizations. Our prototype for Swedish and Finnish uses grammar-level morphology for morpheme segmentation and compact Leipzig-style glosses.

1 Introduction

Automated glossing is the task of deriving morpheme boundaries, lexical glosses, and grammatical labels from object-language text.

We approach this task as parallel generation from a grammatical analysis. Instead of generating a gloss line directly, we parse an object-language sentence into an abstract tree and then linearize it to produce object-language, morpheme-segmented, gloss, and diagnostic English lines.

Here we present a prototype that uses the smart paradigms and language-specific morphology of a custom resource grammar (see §2.2) to process a small corpus of Swedish and Finnish sentences. A small Haskell harness handles tokenization, tree selection, alignment, provenance tracking, structural validation, and report rendering.

2 Background

2.1 Interlinear text

An *interlinear gloss* is a morpheme-by-morpheme annotation of an object-language expression. It can contain, for example, an object-language line, a morpheme-segmented line, and a gloss line with lexical glosses and labels. A free translation line is also commonly included. An *interlinear text* is a sequence of glossed sentences.

We roughly follow the Leipzig Glossing Rules (Comrie et al., 2015): hyphens mark morpheme boundaries, while dots combine multiple metalanguage elements corresponding to one object-language element. Lexical glosses are written in lowercase and labels in small caps, as in PST, SG, or DEF. Ø marks an element with no overt object-language segment. For Swedish, UTR and NEUTR denote common and neuter gender, and parentheses mark inherent gender.

2.2 Grammatical Framework

Grammatical Framework (GF) is both a grammar formalism and a functional programming language. An *abstract syntax module* defines a typed language of *abstract trees*. Its categories are types, and its constructors are typed functions that build well-formed abstract trees. For example, one can form a clause from a noun phrase and a verb phrase with a constructor of type `NP → VP → Cl`. The abstract syntax module leaves the finite verb's position, agreement, and visible inflectional forms unspecified.

A *concrete syntax module* interprets an abstract syntax module by defining a homomorphism from its algebra of abstract trees to a linearization algebra. It assigns each abstract category a linearization type and each abstract constructor a linearization function. Linearization types may be simple *strings*, *records* for discontinuous constituents, or *tables* indexed by parameters such as case, number, gender, person, tense, or polarity. Concrete syntax modules thereby handle word order, agreement, inflection, multiword expressions, and morphophonemic alternations (Ranta and Listenmaa, 2023).

Figure 3 illustrates this structure with excerpts from the Finnish modules. The abstract syntax module declares `water_N` as an abstract `N`. The concrete syntax modules for object-language forms and morpheme segmentation use the same application-level equation, but their aliases refer to parallel resource modules with different linearization types and tables.

Parsing maps a string through a concrete syntax to one or more abstract trees; *linearization* maps an

Det	*var*	*en*	*kall*	*vinter*	*det*	*året.*
det	var	en	kall-Ø	vinter	det	år-et.
EXPL	be.PST	INDF.UTR.SG	cold-UTR.SG	winter(UTR)	DEM.NEUTR.SG	year(NEUTR)-DEF.SG.

'It was a cold winter that year.'

Figure 1: Concrete realization of a sentence from the short story *Pälsen*.

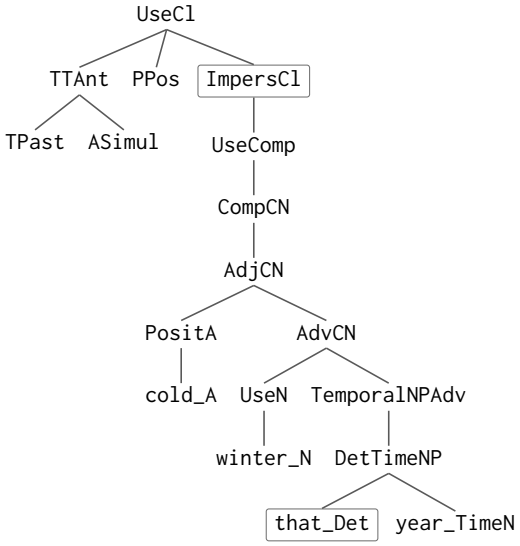

Figure 2: Selected abstract tree for Figure 1. The boxed `ImpersCl` licenses the expletive analysis of the first *det*, while `that_Det` analyses the second as a demonstrative determiner.

abstract tree to a concrete realization. By parsing with one concrete syntax and linearizing the resulting abstract tree with another, we can translate between languages (Ranta, 2011, 2013).

A *resource grammar* in GF is a reusable grammatical layer. GF's Resource Grammar Library (RGL) provides a common grammatical interface for operations such as predication, complementation, modification, and coordination. Each language implements this interface through language-specific syntax and morphology, including inflectional paradigms, agreement, case, word order, and morphophonemic alternations (Ranta, 2009; Ranta and Listenmaa, 2023). For interlinear glossing, we implement the same principles in a custom resource grammar to obtain object-language, morpheme-segmented, and gloss lines from a shared representation.

3 Interlinear glossing with GF

3.1 Introductory example

Figures 1 and 2 show the tool's output for a Swedish example and its selected abstract tree, respectively.

Our tool first asks GF for a bounded set of possible parses, and then uses the Haskell scoring layer to choose the abstract tree used for gloss generation. This tree fixes a particular lexical and grammatical analysis. Parallel concrete syntax modules linearize it to produce object-language, morpheme-segmented, gloss, and diagnostic English lines.

For this sentence, the selected abstract tree analyses the clause as impersonal. This tree implies that the initial *det* is the expletive subject and is glossed as EXPL, while the second *det* is part of the temporal noun phrase *det året* and is glossed as DEM.NEUTR.SG. Thus, while the object-language forms appear identical, we had to recognize that the two occurrences of *det* realize different syntactic rôles. The same tree also analyses *var* as a past active form of *be*, and *en kall vinter* as the complement of the clause.

3.2 Parsing and tree selection

The Haskell harness tokenizes each input sentence and constructs an ordered set of parse variants. It then asks GF for a bounded set of candidates in each *root category*. A *declarative parse* has sentence category `S`, while an *utterance parse* has category `Utt` and can cover wrappers such as punctuation or direct speech. Declarative parses are tried before utterance parses.

For each candidate considered, the harness uses the parallel concrete syntax modules to generate the object-language, morpheme-segmented, and gloss lines. We consider a candidate to be *aligned* when its generated object-language forms match the input words in order and its morpheme and gloss segments correspond.

Within each root category, candidates are ranked by *root-shape penalty*, *grammar-specific penalties*, *GF probability rank*, *tree depth*, and *tree size*, in that order. Lower values are preferred, and later criteria only break earlier ties. The root-shape penalty prefers sentence-shaped roots and known punctuation or direct-speech wrappers over fragmentary roots. The grammar-specific penalties are local priors for recurring overgeneration patterns, such as generic noun-phrase-to-adverb temporal coercions when a more explicit construction is available. GF

Concrete syntax for object-language forms

```
-- KotusLexiconFinCells (K)
vesi_NK : KotusN =
  kotusN27 "vesi" ;

-- KotusMorphologyFinCells (KM)
kotusN27 : Str -> KotusN =
  \lemma -> {s = d27 lemma} ;

-- KukkoJaKanaSourceCellsFin
water_N = KM.useKotusN K.vesi_NK ;
```

Concrete syntax for morpheme segmentation

```
-- KotusLexiconFinHyphens (K)
vesi_NK : KotusN =
  kotusN27 "vesi" ;

-- KotusMorphologyFinHyphens (KM)
kotusN27 : Str -> KotusN = \lemma ->
  let raw : Noun =
    nForms2N (d27 lemma)
  in pairNoun raw (segDArpi raw) ;

-- KukkoJaKanaMorphemesFin
water_N = KM.useKotusN K.vesi_NK ;
```

Concrete syntax of the glossing metalanguage

```
-- ParadigmsFinTags (P)
tagN : Str -> N = \lemma ->
  lin N {
    s = \nf ->
      tagNFormOpen lemma nf ;
    h = Back
    } ;

-- KukkoJaKanaTagsFin
water_N =
  P.tagN "water" ;
```

Figure 3: Parallel concretes for the abstract constant `water_N` of type `N` using Kotus paradigms.

Manifest record: vesi, noun, Kotus type 27 ⟶ *Generated GF call:* `kotusN27 "vesi"`

Syntactic features	**Object-language form**	**Segmentation**	**Gloss**
NOM.SG	*vesi*	*vesi*-Ø	water-NOM.SG
GEN.SG	*veden*	*vede*-n	water-GEN.SG
PART.SG	*vettä*	*vet*-tä	water-PART.SG
ESS.SG	*vetenä*	*vete*-nä	water-ESS.SG

Manifest record: paasi, noun, Kotus type 27 ⟶ *Generated GF call:* `kotusN27 "paasi"`

Syntactic features	**Object-language form**	**Segmentation**	**Gloss**
NOM.SG	*paasi*	*paasi*-Ø	boulder-NOM.SG
GEN.SG	*paaden*	*paade*-n	boulder-GEN.SG
PART.SG	*paatta*	*paat*-ta	boulder-PART.SG
ESS.SG	*paatena*	*paate*-na	boulder-ESS.SG

Figure 4: Parallel components of the concrete realization of Finnish *vesi* ‘water’ and *paasi* ‘boulder’. Selected grammatical features index object-language forms from the RGL-derived type-27 paradigm and the corresponding project-defined segmentation and gloss tables.

probability rank is the position assigned after GF orders the parses for a given parse variant and root category. If all criteria tie, the earlier-enumerated candidate is kept. At most 1,024 candidates for each combination of parse variant and root category are tested for alignment in score order, and the first aligned candidate is selected.

3.3 Parallel morphology and linearization

Once selected, the abstract tree is linearized by parallel concrete syntax modules to produce the object-language forms, their morpheme segmentation, and the corresponding glosses.

GF’s *smart paradigms* construct feature-indexed inflection tables from compact lexical specifications. Kotus (Kotimaisten kielten keskus, the Institute for the Languages of Finland) groups Finnish lexemes into numbered inflection types. Type 27 describes the pattern illustrated by *vesi* (*water*) in Figure 4. A curated lexical manifest records each lemma’s expected Kotus inflection type and, where applicable, consonant-gradation class, as listed in *Nykysuomen sanalista 2024* (Kotimaisten kielten keskus, 2024). A Haskell generator validates each record against the list and emits a GF lexical binding to the corresponding Kotus paradigm, using a call such as `kotusN27 "vesi"`.

The Kotus paradigms were already defined in the Finnish resource grammar, so no new inflectional rules were needed for the object-language forms. For the morpheme-segmentation module, we reused the stems and endings selected by these paradigms and marked their boundaries. This is done once per paradigm rather than separately for each lexeme. For example, the type-27 paradigm gives the stem series *vesi-* : *vede-* : *vet-* : *vete-* for *vesi* and *paasi-* : *paade-* : *paat-* : *paate-* for *paasi*, as shown in Figure 4. The case endings subject to vowel harmony have back- and front-vowel allomorphs, as in *paat-ta* and *vet-tä*, or *paate-na* and *vete-nä*. The existing paradigms already select the appropriate stem and ending; the segmentation module makes the boundary between them explicit and, in plural forms, also separates the plural marker.

The gloss module was also derived from the pre-existing paradigms. Because the glossing metalanguage regularizes the morphology, however, its paradigms could be simplified considerably. Each lexeme contributes a single base gloss, while its inflectional forms are represented by grammatical labels attached with hyphens or dots.

The segmentation and gloss modules are both organized by paradigm, making much of the work systematic. We first spent two hours implementing representative cases by hand; a coding agent then completed the remaining, structurally analogous segmentation and gloss paradigms.

Once the relevant paradigms are in place, adding a regular lexeme does not require changes to the morphology or syntax rules. The lexeme is declared in the abstract syntax and given corresponding entries in the parallel concrete lexicons for object-language forms, morpheme segmentation, and glossing. These entries supply its language-specific inflectional and lexical properties, such as gender, valency, or object case, while the existing syntax rules select and combine the appropriate forms in each linearization.

3.4 Diagnostic translation

The selected abstract tree can also be linearized in other languages. This follows the usual GF pattern of simply adding another concrete syntax module. We use an English concrete syntax module to obtain a diagnostic translation. The English realization may use a different construction from that used in the object language. For example, Swedish *verkställande direktör* contains a present participle (lit. *executing director*) but is rendered in English as *executive director*. We model this via a lexicalized compound noun `executive_director_CN`. This diagnostic translation is not used in the scoring and glossing machinery. We include it only as an approximation to the free translation sometimes given in interlinear glosses.

3.5 Alignment check and fallback

Only an aligned candidate yields a gloss based on a full grammatical analysis. If none of the candidates tested for alignment succeeds but full parses are available, a separate fallback step selects a parse according to the parse-variant, root-category, and scoring priorities described above. This parse is used for the diagnostic English translation, while the gloss is produced by lexical analysis. If no full parse is available, glossing proceeds directly from lexical analysis. In either fallback case, GF morphology is used where possible, and the remaining object-language token groups are marked as unanalysed.

3.6 Coverage

The current lexica cover the first 32 sentences (493 words) of the Swedish short story *Pälsen* and all 20 sentences (151 words) of the Finnish folk tale *Kukko ja kana*.

4 Related work

Statistical and neural methods have been applied to automated glossing in low-resource language-documentation corpora (Samardžić et al., 2015; Moeller and Hulden, 2018; Zhao et al., 2020; Ginn et al., 2023; Okabe and Yvon, 2023). Their prediction targets range from labels assigned to characters or morphemes to complete gloss sequences, with morphological segmentation either supplied or inferred. Recent work explores multilingual pretraining for interlinear glossing (Ginn et al., 2024), prompting with large language models (Yang et al., 2024; Elsner and Liu, 2025), and additional information from translations and dictionaries (Yang et al., 2024).

An earlier grammar-backed point of comparison is finite-state morphological glossing. Antworth (1992) uses PC-KIMMO, KTEXT, and ITF to produce automatically glossed interlinear text, and Smith and Hulden (2016) present a later lexc/foma analyser, lemmatizer, and glosser for Sahidic Coptic. These systems derive glosses from word-level morphological analyses and retain multiple readings when a form is ambiguous, leaving disambiguation to the user or later processing. By contrast, in our Swedish example, sentence-level analysis distinguishes the expletive and demonstrative uses of *det*.

5 Conclusion

We have approached interlinear glossing as parallel generation from a grammatical analysis. Parallel concrete syntaxes linearize the selected abstract tree to produce object-language forms, morpheme segmentation, glosses, and a diagnostic English translation. The glosses reflect the sentence-level analysis. For example, Swedish *det* receives different labels in its expletive and demonstrative uses. The current lexica cover 32 Swedish and 20 Finnish sentences. The next step is to extend this coverage to further texts and determine how much of the abstract syntax and paradigm-level morphology can be reused.

Limitations

The implementation is an experimental prototype. The grammars are compositional and reusable, but the lexica and tree-selection heuristics are over-specialized to the selected examples. Scaling will therefore require larger lexica and more general tree selection. We have also not yet evaluated our tool on larger corpora.

Acknowledgements

This work was supported by the EU through the ERC grant *NextReason* (doi:10.3030/101200949).

We thank the Institute for the Languages of Finland (Kotus) for making *Nykysuomen sanalista* and its inflectional data available under an open license. We also thank the anonymous reviewers for their helpful comments.

References

Evan L. Antworth. 1992. Glossing text with the PC-KIMMO morphological parser. *Computers and the Humanities*, 26:389–398.

Bernard Comrie, Martin Haspelmath, and Balthasar Bickel. 2015. The Leipzig glossing rules: Conventions for interlinear morpheme-by-morpheme glosses. Revised version of February 2008; last change May 31, 2015.

Micha Elsner and David Liu. 2025. Prompt and circumstance: A word-by-word LLM prompting approach to interlinear glossing for low-resource languages. In *Proceedings of the 22nd SIGMORPHON workshop on Computational Morphology, Phonology, and Phonetics*, pages 1–14, Albuquerque, New Mexico, USA. Association for Computational Linguistics.

Michael Ginn, Sarah Moeller, Alexis Palmer, Anna Stacey, Garrett Nicolai, Mans Hulden, and Miikka Silfverberg. 2023. Findings of the SIGMORPHON 2023 shared task on interlinear glossing. In *Proceedings of the 20th SIGMORPHON workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 186–201, Toronto, Canada. Association for Computational Linguistics.

Michael Ginn, Lindia Tjuatja, Taiqi He, Enora Rice, Graham Neubig, Alexis Palmer, and Lori Levin. 2024. GlossLM: A massively multilingual corpus and pretrained model for interlinear glossed text. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12267–12286, Miami, Florida, USA. Association for Computational Linguistics.

Kotimaisten kielten keskus. 2024. Nykysuomen sanalista. Dataset. Updated 11 April 2025. Licensed under CC BY 4.0.

Sarah Moeller and Mans Hulden. 2018. Automatic glossing in a low-resource setting for language documentation. In *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*, pages 84–93, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shu Okabe and François Yvon. 2023. Towards multilingual interlinear morphological glossing. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5958–5971, Singapore. Association for Computational Linguistics.

Aarne Ranta. 2009. The GF resource grammar library. *Linguistic Issues in Language Technology*, 2(2):1–63.

Aarne Ranta. 2011. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, Stanford.

Aarne Ranta. 2013. Grammatical Framework tutorial. Third GF Summer School.

Aarne Ranta and Inari Listenmaa. 2023. Grammatical Framework: From interlingual semantics to morphophonemic processes. In Arvi Hurskainen, Kimmo Koskenniemi, and Tommi Pirinen, editors, *Rule-Based Language Technology*, volume 2 of *NEALT Monograph Series*, pages 9–48. Northern European Association for Language Technology.

Tanja Samardžić, Robert Schikowski, and Sabine Stoll. 2015. Automatic interlinear glossing as two-level sequence classification. In *Proceedings of the 9th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH)*, pages 68–72, Beijing, China. Association for Computational Linguistics.

Daniel E. Smith and Mans Hulden. 2016. Morphological analysis of Sahidic Coptic for automatic glossing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2584–2588, Portorož, Slovenia. European Language Resources Association (ELRA).

Changbing Yang, Garrett Nicolai, and Miikka Silfverberg. 2024. Multiple Sources are Better Than One: Incorporating External Knowledge in Low-Resource Glossing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4537–4552, Miami, Florida, USA. Association for Computational Linguistics.

Xingyuan Zhao, Satoru Ozaki, Antonios Anastasopoulos, Graham Neubig, and Lori Levin. 2020. Automatic interlinear glossing for under-resourced languages leveraging translations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5397–5408, Barcelona, Spain (Online). International Committee on Computational Linguistics.