French Amalgam: A machine-learned sentence realization system

Martine Smets, Michael Gamon, Simon Corston-Oliver and Eric Ringger
Microsoft Research
One Microsoft Way
Redmond, WA 98052, U.S.A.

{martines, mgamon, simonco, ringger}@microsoft.com

Résumé – Abstract

Cette communication présente la version pour le français d'Amalgam, un système de réalisation automatique de phrases. Deux des modèles du système sont décrits en détail, et nous expliquons comment la performance des modèles peut être améliorée en combinant connaissances et intuition linguistiques et méthodes statistiques.

This paper presents the French implementation of Amalgam, a machine-learned sentence realization system. It presents in some detail two of the machine-learned models employed in Amalgam and shows how linguistic intuition and knowledge can be combined with statistical techniques to improve the performance of the models.

Keywords – Mots Clés

Réalisation de phrase, génération automatique, arbres de décision, français.
Sentence realization, generation, machine-learning, decision trees, French.

1 Introduction

Amalgam is a multilingual sentence realization system. Developed originally for German (Corston-Oliver et al. 2002, Gamon et al. 2002), it has been adapted to French (Smets et al. 2003). Amalgam maps a representation of propositional content (henceforth "logical form") to a surface syntax tree via intermediate syntactic representations. The mappings are performed with linguistic operations, the context for which is primarily machine-learned. The leaf nodes of the resulting syntax tree contain all necessary information from which to generate the surface string.

Other systems use machine learning techniques for sentence realization in generation. The Nitrogen system, for example, uses a word bigram language model to score and rank a large set of alternative sentence realizations (Langkilde and Knight, 1998a, 1998b). Other recent

approaches use syntactic representations: FERGUS (Bangalore and Rambow 2000) and Halogen (Langkilde 2000, Langkilde-Geary 2002) use syntax trees as an intermediate representation to determine the optimal string output.

The adaptation of German Amalgam to French has been discussed elsewhere (Smets et al. 2003). In this paper, we discuss French Amalgam in some detail by presenting two of the machine-learned models and a tool which allows the researcher to manually inspect the relevant training data for each model. In this way, the researcher can understand why the model makes the decisions it makes and can improve the model by adding relevant linguistic features. The researcher can thus leverage the richness of the linguistic information to optimize the models.

2 French Amalgam

Amalgam takes as its input a logical form graph, i.e., a sentence-level dependency graph with fixed lexical choices for content words. Function words are represented as features or attributes (c.f. Heidorn 2000). The logical form represents the predicate-argument structure of the sentence and also includes some semantic information (e.g., the meaning of time and location prepositions). During sentence realization, the logical form is first degraphed into a tree and then augmented by the insertion of function words (determiners, auxiliaries, some prepositions, etc.) and syntactic labels. Linguistic operations such as the introduction of coordination, raising, ordering, aggregation, punctuation, inflection, etc. are performed to produce a surface syntax tree. The linguistic contexts for insertions and all other operations are learned by the WinMine toolkit (Chickering 2002) and represented in decision tree models. Finally, an output string is read off the leaf nodes.

2.1 Stages of the system

Amalgam includes eight stages. The first stage involves language-neutral transformations from a graph representation to a tree representation. Further stages process that tree representation until it results in a surface syntax tree. The contexts for most linguistic operations are machine-learned. The only contexts that are not machine-learned, are those for which there is not sufficient data to train robust models.

Stage 1 Pre-processing (procedural)
- degraphing of the semantic representation
- retrieval of lexical information

Stage 2 Flesh-Out (machine learned):
- assignment of syntactic labels
- insertion of function words
- insertion of clitics
- assignment of case (rule-based)

Stage 3 Conversion to syntax tree:
- introduction of syntactic representation for coordination (procedural)
- head-switching (machine learned)

Stage 4 Movement**:**
- raising, wh-movement (rule-based)

Stage 5 Ordering (machine learned):
- ordering of constituents and leaf nodes in the tree

Stage 6 Surface cleanup (machine learned):
- lexical choice of determiners and relative pronouns
- syntactic aggregation

Stage 7 Punctuation (machine learned)
Stage 8 Inflectional generation (rule-based)

In the logical form, certain constituents that occur in the surface string are represented as features on nodes. For example, the French definite articles *le/la/les* are represented in the logical form by the feature [+Def] on a lexical node. Similarly, negation is represented by the feature [+Neg] on the constituent it modifies. In converting from a logical form to a surface-oriented representation (Stage 2, Flesh Out), these features are converted to independent constituents.

The head-switching module in stage 3 permutes the head and one of its children in configurations where the semantic head is not the same as the syntactic head. In French Amalgam, this occurs with partitive constructions and with modals. Modals in French behave syntactically like main verbs and govern an infinitival clause, but semantically, the modal and the verb it modifies express a single proposition. In our logical form, modals are represented as children of the node that they govern in syntax. In Figure 1, the modal, *pouvoir*, is an attribute of the verb which in syntax is its complement *utiliser* ('use') (1). Features such as tense, mood, and negation are copied onto the semantic head. In the conversion from logical form to syntactic tree in stage 3, the modal becomes the head of the sentence in (1) and the infinitive, its child. This is performed by the head-switching module.

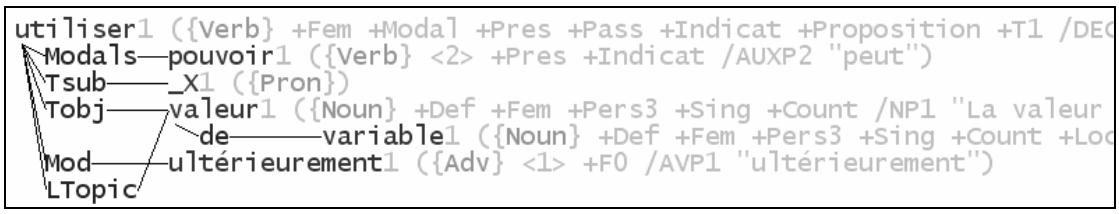

Figure 1: Logical form for sentence (1)

(1)	La	valeur	de	la	variable	peut	être	utilisée	ultérieurement
	'the	value	of	the	variable	can	be	used	later'

Later stages introduce coordination, order constituents and "clean up" the syntax tree. Cleaning up (stage 6) determines the surface form of determiners and relative pronouns and aggregates duplicated material in coordinated structures.

There are a total of eighteen decision trees employed in the pipeline, and the complexity of the decision trees varies with the complexity of the modeled task. The number of branching nodes in the decision tree models in French Amalgam ranges from ten for a relatively simple task, such as determining the context for the insertion of the subordinate conjunctions *que* and *si*, to 1016 for the more difficult task of determining the label of a constituent. The ordering model stands apart from the others, with 4,536 branching nodes (for details see Ringger et al. (in preparation)).

2.2 Data and feature extraction

The training data for all the models consist of a set of 100,000 sentences drawn from software manuals. The sentences are analyzed in the NLPWin system, which provides a syntactic and logical form analysis (Heidorn 2000; Gamon *et al*. 1997). Nodes in the logical form representation are linked to the corresponding syntactic nodes, allowing us to learn contexts for the mapping from the logical form representation to a surface syntax tree. The data is split 70/30 for training versus model parameter tuning. For each set of data we build decision trees at several different levels of granularity (by manipulating the prior probability of tree structures to favor simpler structures) and select the model with the maximal accuracy as determined on the parameter tuning set. We attempt to standardize as much as possible the set of features to be extracted by exploiting the full set of features and attributes available in the analysis, instead of pre-determining a small set of potentially relevant features. This allows us to share the majority of code between the individual feature extraction tasks (and among implementations for different languages). Typically, we extract the full set of available analysis features from the node under investigation, its parent and its grandparent. This provides us with a sufficiently large structural context for the operations.

For most of the models we also add a small set of features that we believe to be important for the task at hand, and that cannot easily be expressed as a combination of analysis features or attributes on constituents. In this way, we can exploit linguistic information which we know is relevant to improve the accuracy of our models. For example, the model which inserts negation must choose between inserting "*ne pas*" or "*ne*". The first value is the default as in (2), while the second value is chosen if a negative quantifier figures among the arguments of the verb, as in (3).

(2)

Assurez-vous	que	les	périphériques	ne	bougent	pas	ou	ne	vibrent	pas
Assure you	that	the	devices	*neg*	move	*neg*	or	*neg*	vibrate	*neg*

"Make sure the devices are not moving or vibrating."

(3)

Dans	ce	cas,	aucune	modification	n'	est	observée
In	this	case,	no	change	*neg*	is	observed

"In this case, no change is seen"

In order to learn the correct context for each form of the negation, the decision tree has to take into account the presence of negative quantifiers in the clause. We define specific functions to compute that feature.

Linguistic intuition in a vacuum is not always sufficient to determine which features to use to obtain an accurate model for a certain set of data. It can be very useful to look at the way the training data was classified by the decision tree classifier to understand what features led to correct or incorrect classification. A failure analysis tool allows us to inspect the sentences of the training corpus corresponding to each leaf of the decision tree. Clicking on a leaf node displays an HTML page giving detailed failure analysis information: the values of the features used in the classification, the data correctly classified, and the data incorrectly classified. This allows the researcher to investigate which features were used by the classifier, and why incorrect classifications were made. Seeing the misclassifications suggests new features that could be added to discriminate the problematic cases, and thus help improve the accuracy of

the model. Looking at data correctly classified can enable us to discover new linguistically interesting and/or domain-specific generalizations from the data.

3 Examples of models

3.1 Infinitive markers

Infinitive markers, like most function words, are not represented as semantic nodes in our logical forms, and need to be inserted during sentence realization. The choice of infinitive marker depends on the subcategorization features of the lexical item governing the infinitive, but often a word subcategorizes for more than one infinitive marker. For example, *essayer* can be followed by "*de infinitive*" but is followed by "*à infinitive*" if it is reflexive. Also, nouns and adjectives can introduce an infinitive clause, but do not always subcategorize for a specific infinitive marker. Thus, the model needs to capture the contexts for each infinitive marker. There are four possible values: *à*, *de*, *pour* and *none*.

3.1.1 Features in the infinitive marker model

For each data point, 272 features were extracted. Of these 272 features available to the decision tree learner, 28 were selected as having predictive value for the model. The selected features fall into the following categories (in decreasing order of importance, according to the learner):

- Grammatical function of the infinitive clause, e.g., whether it is a purpose clause or an object.
- Is there already a governing preposition?
- Subcategorization features of the parent
- Category of the parent
- Semantic features of the node, parent and grandparent
- Subcategorization features of the node itself
- Other arguments of the parent
- Is there a preposition introducing the parent?
- Function of the parent
- Nominal features of the parent
- Arguments and agreement features of the grandparent

The top features selected by the decision tree learner correspond to linguistic intuition: the choice of infinitive marker depends on the function of the clause, on whether the infinitive clause is already introduced by a preposition, and on subcategorization features of the parent. However, it might seem surprising that a preposition introducing the node's parent would be relevant. A look at the training data explains why this is the case. In the logical form, a modal is represented as a child of the verb it syntactically governs. However, in syntax, the modal behaves like other verbs and can be the location of insertion of function words. Thus, when the node being considered is a modal, it is relevant to know whether its parent already has a preposition as an attribute. An example is given in (4) and Figure 2. The verb *utiliser* "use" is

already introduced by the preposition *avant de* "before", and, as the decision tree classifier has learned, no preposition thus needs to be inserted on *pouvoir* "be able".

(4)	Avant	de	pouvoir	utiliser	une	DLL	de	balise	active,
	Before	of	be able	use	a	DLL	of	tag	active
	celle-ci	doit	être	enregistrée	sur	le	système		
	this one	must	be	registered	on	the	system'		

"Before you can use any Smart Tag DLL, it must be registered on the system."

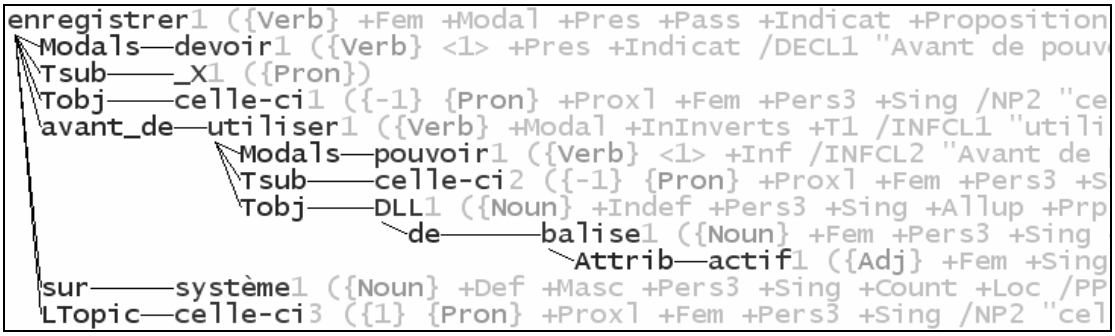

Figure 2 Logical form for sentence (4)

Finally, infinitival clauses can be governed by verbs, adjectives or nouns. The category of the parent and nominal features of the parent are thus relevant in the choice of a preposition.

Examples of infinitival clausal complements of a noun are given in (5). The model needs to learn which preposition, *à, de* or *pour* to insert in which environment.

(5a) La première étape pour obtenir un certificat de serveur consiste à créer le fichier de demande
"The first step you will need to perform to get a server certificate is creating the request file."

(5b) Pour plus d'informations sur la procédure à suivre pour que les applications tirent profit de cette fonctionnalité, consultez le site Web suivant :
"For more information about writing applications to take advantage of this feature, visit the following Web site:"

(5c) Ceci est utile si vous avez l'intention de sauvegarder le fichier .pst sur une disquette
"This is useful if you plan to back up the .pst file to a floppy disk"

3.1.2 The infinitive marker model

This model has 84 branching nodes. The precision, recall and F-measures (the harmonic mean) of this model are given in Table 1. The overall accuracy of this model is 0.9315, compared to a baseline of 0.4024 if the model were to always assign the most frequent value, *pour*.

Value	Precision	Recall	F-measure
none	0.9727	0.9698	0.9712
de	0.8755	0.9391	0.9062
pour	0.9673	0.9585	0.9629
à	0.8164	0.6865	0.7459
overall accuracy	0.9315		

Table 1: Precision and recall for the infinitive marker model

3.2 Auxiliaries

Auxiliaries do not have corresponding lexical nodes in the logical form. Instead they are represented as features. During the conversion from logical form to a surface-tree, the auxiliaries must be inserted. For example, [+Perf] (perfect) is linked to tense auxiliaries, [+Pass] (passive) to the passive auxiliary. In French, there are two tense auxiliaries, *avoir* and *être.* The selection of the tense auxiliary is determined by the specific verb. However, reflexive verbs always utilize *être* as a tense auxiliary, and the passive voice always uses *avoir* as a tense auxiliary. Also, some verbs can occur with either auxiliary, depending on whether they are used transitively or intransitively, as in (6).

(6a) Il est sorti
'he is gone out'
"He has gone out ." (Intransitive)

(6b) Il a sorti la voiture
'he has taken out the car'
"He has taken the car out."

For passive perfect verbs, both auxiliaries need to be inserted. Finally, when verbs are introduced by a modal, the auxiliary is inserted on the modal, not on the main verb. These different configurations need to be learned and represented in the model in order to accurately predict auxiliary insertion. The values from which the model chooses are: *être*, *avoir*, *être-avoir*, *rester*, *avoir-mod*, *none.*

3.2.1 Features of the auxiliary model

During training, 245 features are extracted for each data point, but only 16 are determined by the decision tree learner to be predictive. These features include linguistically intuitive features such as [+Etreaux] (i.e., a lexical feature on a verb indicating that it takes *être* in the periphrastic past tense construction, the *passé composé*), the passive feature [+Pass] and the tense feature [+Perf] (indicating a *temps composé,* a perfect construction). The model also looks at the modal of a verb, and checks whether the modal has the feature [+Perf] (which indicates that an auxiliary should be inserted on the modal). The types of features used by the model are listed below, in decreasing order of importance according to the decision tree learner.

- Passive feature

- Tense and aspect features (of verb and modals)
- Syntactic category
- Arguments of the verb (subject and object)
- Subcategorization features
- Etreaux
- Negation
- Arguments of parents (object)
- Presence of preposition introducing the verb

The learner thus does an excellent job at discovering linguistically relevant features. Some features selected in the model are surprising, however. For example, the negation feature (*Neg*) influences the realization of an auxiliary in the following way:

- If the node is passive, past, the head of an infinitive clause, and has the feature [+Neg], then the most likely auxiliary is *être*, with p=0.857
- If the node is passive, past, the head of an infinitive clause and is not [+Neg], then the most likely auxiliary is *être-avoir*, with p=0.605.

Although this is counter-intuitive, it is actually true, for the first context, of every single sentence in the test set of the training corpus. This is the situation where a passive verb is the complement of modal *pouvoir* and bears the passive auxiliary while the modal carries the tense inflection (example in (7)).

(7) La connexion à Internet n' a pas pu être effectuée ou
'the connection to Internet *neg* has not can be performed or

une erreur de connexion est survenue
an error of connection is occurred

"Could not connect to the Internet, or the Internet connection returned an error."

When there is no negation, the prediction is confirmed in most cases, but there are exceptions. A positive example is given in (8). The tense and passive auxiliaries are on the main verb.

(8) Money renvoie un message vous signalant qu' un transfert associé
Money sends a message you signaling that a transfer associated

à un placement du fichier importé peut avoir été ignoré
to a investment of the file imported may have been ignored

"Money returns a message alerting you that some transfer associated with some investments in the file you imported may have been ignored."

The example in (9) goes against the prediction of the model: the tense auxiliary is on the modal, although there is no negation.

(9) La base de données a été réparée, certaines données ont pu être perdues
'the database has been repaired, some data have may be lost'
"Database repaired, some data may have been lost"

One aspect of the problem is that the model does not have enough information to distinguish between *peut avoir été ignoré* ('may have been ignored') and *a pu être ignoré* ('could have been ignored'). In the first context, the main verb carries both auxiliaries; in the other context,

the tense auxiliary is on the modal, the passive auxiliary on the main verb. The verb features available to the model are identical in both cases (features are copied from the modal onto its parent in the logical form), and it relies on other information to distinguish between these two cases. This information is the negation feature, an unlikely candidate to distinguish uses of auxiliaries. There are two interesting aspects of this. First, the decision tree needs more information to distinguish between both contexts, and by looking at the training data, we can easily determine which features to add to the set of extracted features. Second, quite surprisingly, negation is successful in predicting the insertion of the correct auxiliary in most cases. Interestingly, the model has classified the use of the modal *pouvoir* in two semantic categories: the non-epistemic meaning (*be able to*) and the epistemic meaning. All instances in the negative context have the first meaning, while all but one instance in the positive context have the epistemic meaning. It seems that, in this particular corpus at least, *pouvoir* in the present tense followed by a past infinitive tends to have the epistemic reading, while it has the non-epistemic meaning when it is inflected for tense. At the same time, again in this particular corpus, negation on the modal tends to occur with the non-epistemic reading.

3.2.2 The auxiliary model

This model has 42 branching nodes. The precision, recall and F-measure are given below. The baseline was 0.9132 corresponding to most common value: 'none'. The scores for *rester* "remain" indicate data sparsity: there are not enough cases in the data with *rester* used as an auxiliary.

Value	Precision	Recall	F-measure
none	0.9998	0.9997	0.9997
être-avoir	0.9807	0.9251	0.9521
avoir-mod	0.9688	1.0000	0.9841
avoir	0.9487	0.9906	0.9692
être	0.9867	0.9867	0.9867
rester	0.0000	0.0000	0.0000
overall accuracy	0.9979		

Table 2: Precision and recall of the auxiliary model

4 Conclusion

We have presented French Amalgam, a sentence realization system for French. It takes as input a logical form and outputs a surface syntax tree after a series of linguistic operations, the contexts for which are machine-learned. We have presented two of the machine-learned models of French Amalgam and have discussed features selected by these models in some detail. This discussion was made possible by our failure analysis tool, which allows the researcher to examine the data classified by the decision tree and debug incorrect classifications. Inspection of the training data leads to the discovery of interesting generalizations about the corpus. Moreover, it allows statistical techniques to be combined with linguistic knowledge to improve the performance of the system.

References

Bangalore S. and Rambow O. (2000). Exploiting a probabilistic hierarchical model for generation. In *Proceedings of COLING 2000*, Saarbrücken, Germany, pp. 42-48.

Chickering D. M. (2002) *The WinMine Toolkit*. Microsoft Technical Report 2002-103.

Corston-Oliver S., Gamon M., Ringger E. and Moore R. (2002) *An overview of Amalgam: a machine-learned generation module.* In "Proceedings of the International Language Generation Conference 2002", New York, pp. 33-40.

Gamon M., Ringger E., Corston-Oliver S., Moore R. (2002): *Machine-learned contexts for linguistic operations in German sentence realization.* In: "Proceedings of ACL 2002", pp. 25-32.

Gamon, M., Lozano, C., Pinkham, J. and Reutter, T. (1997) "Practical experience with grammar sharing in multilingual NLP". In Burstein J., Leacock C., eds, *Proceedings of the Workshop on Making NLP Work*, ACL Conference, Madrid, Spain.

Heidorn, G. (2000) Intelligent writing assistance. In Dale R., Moisl H. and Somers H. (eds). *A Handbook of Natural Language Processing: Techniques and Applications for the Processing of Language as Text*. Marcel Dekker, New York

Langkilde I. (2000): *Forest-Based Statistical Sentence generation.* In: "Proceedings of NAACL 2000", pp. 170-177.

Langkilde-Geary I. (2002) *An Empirical Verification of Coverage and Correctness for a General-Purpose Sentence Generator.* In "Proceedings of the International Language Generation Conference 2002", New York, pp. 17-24.

Langkilde I. and Knight K. (1998a). The practical value of n-grams in generation. *Proceedings of the 9th International Workshop on Natural Language Generation, Niagara-on-the-Lake, Canada*. pp. 248-255.

Langkilde I. and Knight K. (1998b). Generation that exploits corpus-based statistical knowledge. *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics* (COLING-ACL 1998). Montréal, Québec, Canada. 704-710.

Palmer F. (1986) *Mood and Modality*, Cambridge University Press, Cambridge.

Ringger E., Gamon M., Smets M., Corston-Oliver S. and Moore R. (in preparation) Linguistically informed models of constituent structure for ordering in sentence realization.

Smets M., Gamon M., Corston-Oliver S. and Ringger E. (2003) The adaptation of a machine-learned sentence realization system to French, to appear in *Proceedings of the 10th conference of the European Chapter of the Association for Computational Linguistics*, 2003.