



# Automatic Summarization of Scientific Papers

Robin Cosbey & Josh Loehr



# Motivation

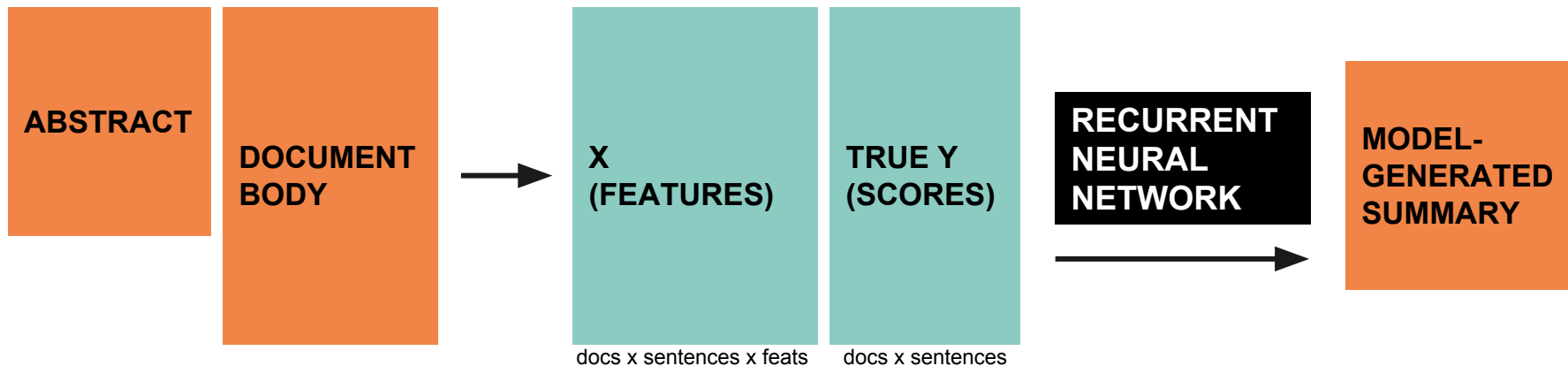
With a growing number of scientific papers, having access to concise summaries can help researchers.

Generate **extractive** summaries of scientific articles.

The goal is to build a model which can create summaries resembling the papers' abstracts, without having knowledge of the abstract itself.

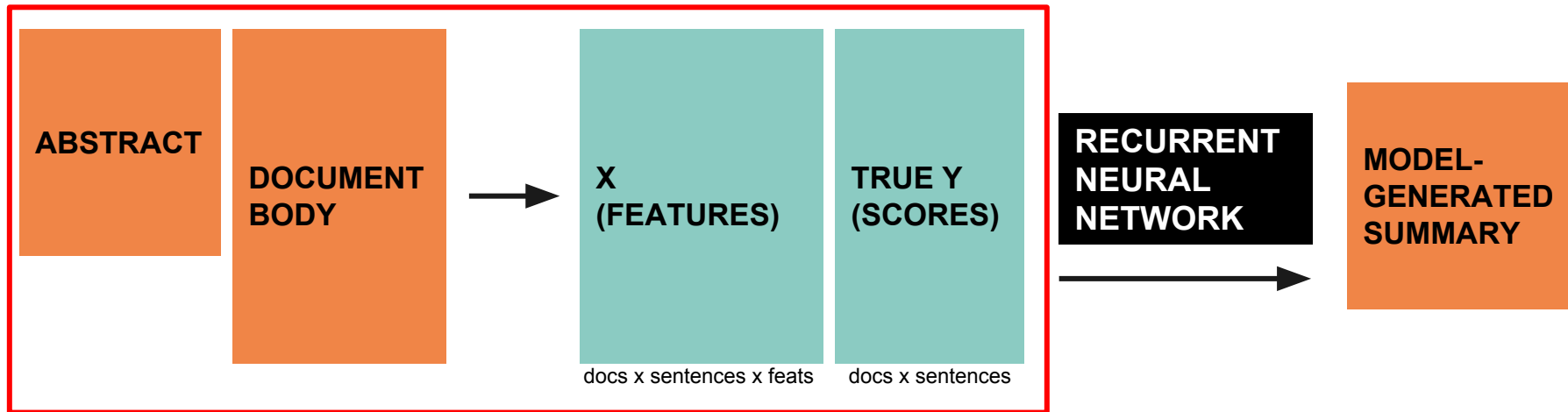
# Approach

- Extraction-based
- Single document



# Approach

- Extraction-based
- Single document





# Datasets

- **Computation and Language corpus**
  - 183 scientific articles from Association for Computational Linguistics conferences
  - Documents automatically converted from LaTeX to XML
- **MEDLINE/PubMed corpus**
  - ~ 1.8 million documents available for free use
  - Documents converted to XML with specific MEDLINE/PubMed data elements



# Preprocessing

- Download the documents
- Scrape XML files (-> abstract.txt, article.txt)
  - Extract <abstract> and <body> tag contents - leave out e.g. references, footnotes, acknowledgements
  - Convert XML tags to special tokens ({{HED}}, {{REF}}, {{EQN}})
- Parse sentences (-> \*.sentences)
  - Custom boundary detection using regular expressions
  - E.g. `text = re.sub(r'(\s+(?!{{HED}}).*)\s+({{HED}})', '\g<1>. {{HED}}', text)`
- Word Tokenization (-> \*.tokens)
  - Replace numerics ({{NUM}}), symbols ({{SYM}})
  - Remove stopwords
  - Stemming



# Training Data

- Features
  - Build a feature vector for each sentence in each document
  - Replace tokens with token IDs, grouping singletons together
  - Vector = [header\_id] + [token\_ids]
  - Can easily extend to include more features: journal\_id, cue words, TF-IDF values, etc.
- Labels
  - TF-IDF vectorize each sentence in abstracts and articles
  - Compare article sentences with abstract sentences
  - Score each article sentence as maximum cosine similarity with abstract sentences
  - Rank and order scores by most similar to least, replace with sentence indices
- 80/10/10 split

# Preprocessing

```
<ITEM>MOOD expresses sentence modalities including
a specification of
which constituents to topicalize in a German declarative
</ITEM>
<ITEM>The predicate argument structure is reflected by
features; ARGUS contains a list of arguments.
</ITEM>
<ITEM>Different sorts of free temporal and local adjuncts
specified by corresponding features. In Figure <REF/>
it is represented under TIME-ADJ.
</ITEM>
<ITEM>Arguments and, in part, adjuncts are specified for
cardinality, for quantificational force (under CONTENT)
and further details such as name strings and natural gender
</ITEM>
<ITEM>Temporal adjuncts relate to some context (e.g. they
are indexical (e.g. on Wednesday, February 7, 1996). A
common combinations in German are covered.
</ITEM>
</P>
</DIV>
<DIV ID="3" DEPTH="1" R-NO="3"><HEADER> The Template
(TGL) </HEADER>
<P>
TGL defines a general format for expressing production
precondition-action pairs (cf. Figure <REF/>). A TGL
is applied if its preconditions are met. A TGL rule is successful
if the action part has been executed without
failure to apply a TGL rule signals that the
rule does not cover the portion of the input structure
</P>
<P>
Figure <REF/> shows a sample TGL rule. It corresponds
to a verb covering
a direct object, an optional temporal adjunct, an optional
for a duration (such as for an hour),
an optional local adjunct (such as at the DFKI building)
infinite verb form.
Given the input GIL structure of Figure <REF/>, the
rule can be generated from this rule. Among the optional
only the temporal adjunct would find appropriate material
in the input structure (under THEME.TIME-ADJ).
</P>
```

9605010.xml

```
<ITEM>The current work in surface realization of
natural, abstract algorithms that interpret declarative
grammars. It is claimed that this way,
parsing and generation, or a generator can interpret
(e.g. in machine translation). A prominent example
algorithm is semantic-head-driven generation
used with HPSG, CCG, DCG and several other formalisms
of surface realization has several drawbacks.
Pars have been developed with parsing as the primary
mind. Adapting their semantics layer to a general
achieving reversibility, can turn out to be a difficult
task. Second, many linguistically motivated grammars
of information presentation, such as fill-in-the-blanks,
or semi-frozen formulae used for greeting,
grammar-based logical form representation hardly
face to deep generation processes. Grammar-based
systems, a compositional reflex of the syntactic
structure too closely to the surface form to be generated.
Only little attention has been paid to interface
adequately to deep generation processes, e.g. by
fluently the order of results of the former. This
intended in this contribution, overcomes many flaws
realization systems that arise in concrete applications.
It can be smoothly integrated with 'deep' generation
of scanned text, templates, and context-free rules.
It allows for both textual and tabular output, efficient
strings for additional solutions, and can be adapted to
linguistic properties (regarding style, grammar, etc.).
TG/2 is based on restricted production systems and
modularity of processing and linguistic knowledge.
transparent and reusable for various applications.
have been used both for modeling human thought (e.g.
construction of knowledge-based expert systems) and
the use of the modularity gained by separating the
generation, production systems have disappeared from
because of their limited transparency caused by
effects. In particular, side effects could modify
that other rules become applicable [ <REF/> ].
action pairs can be used in a more restricted way
disallowing side effects that affect the data.
are tests over the database contents (the generated
and actions typically lead to a new subset of the
```

article.txt

```
<ITEM>The current work surface realize concentr use general abstract
ret declar defin non direct grammar
it claim way grammar reus pars generat generat interp
machin translat
a promin exampl type abstract algorithm senant head d
use <{acr}> <{acr}> <{acr}> sever formal
in practice type surfac realize sever drawback
first man exist grammar develop pars primari type pr
adapt senant layer generat algorithm thus achiev reve
erprits <{ref}>
second main linguist motiv grammar cover common mean
abl bullet list sent frozen formula use greet letter
final grammar base logic form represent hard serv syst
nerat process
grammar base senant larg extent compos reflex synta
espond close surfac form generat
as consequ littl attent paid interfaz type realize ad
ess eg allow latter influenc order result former
the system <{acr}> <{num}> present contribut overcom
e surfac realize system aris concret applic
In particular <{acr}> <{num}> smooth integr deep gene
n text templat context free rule singl formal allow t
effici reus generat subnat addit solut parameter acc
regard style grammar fine grain rhetor etc
<{acr}> <{num}> base restrict product system techniqu
ess linguist knowledge henc make system transpar reusa
product system use model human thought eg <{ref}> con
expert system eg <{ref}>
In spite modular gain separ rule bas interpret produ
ocus current research limit transpar caus various type
in particular side effect could modifi data base way
erff)
honey precondition action pair use restrict way preserv
e effect action databas
in <{acr}> <{num}> precondition test databas content gen
ction type lead new subset rule applic would test se
by constrain effect product rule way disadvantage earl
d
at time consider flexibl maintan regard linguist kno
a product rule may involv direct nap surfac form can
portion surfac text templat induc applic rule classif
templat base generat method correct critic bec inflex
ic rhetor demand man applic
on hand templat success use demand could hard wire ru
in <{acr}> <{num}> rule writer choose degree abstract a
she freell intermix kind rule
the rest paper organ follow
<{acr}> <{num}> assum input predic argument structur
```

article.tokens

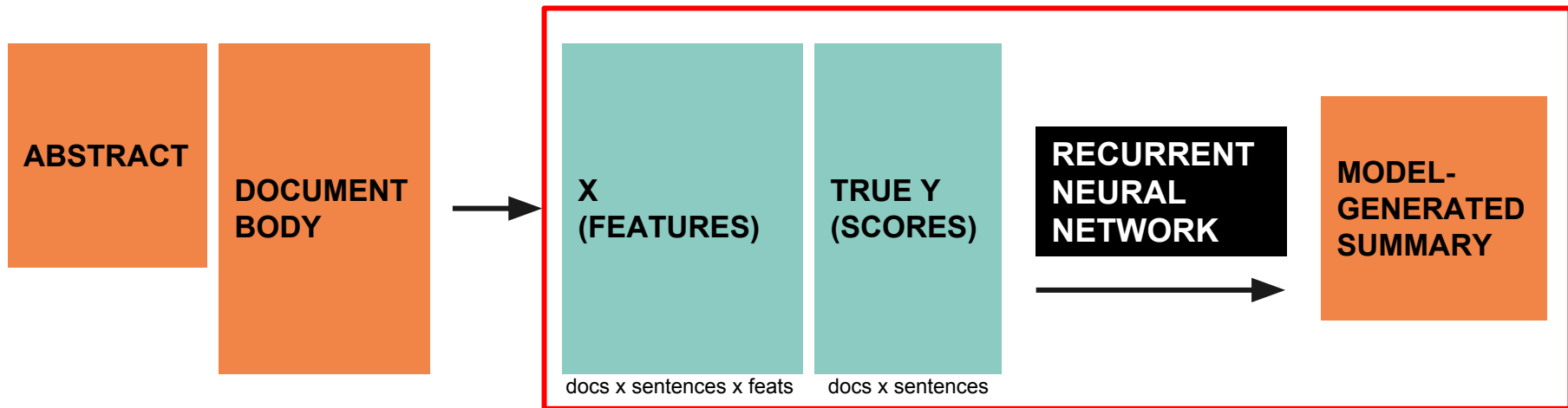
```
0 286 46 131 1214 1046 50 769 1202 74 213 157 853 23 2
0 64 1310 381 69 2000 319 275 275 213 682 69 32 1088 68
0 111 2351 162 236 1202 74 885 824 825 275 6 50 117 117
0 200 929 236 131 1214 694 2117
0 234 616 741 69 3 319 2541 236 186 465
0 558 885 4996 275 74 856 1364 262 899 896 1500 6
0 198 616 89 842 69 434 570 21 572 73 3161 403 2943 88
105
0 829 69 168 888 99 268 627 561 426 1474 1171 275 186
0 69 168 885 48 1665 378 391 118 146 365 589 958 131 9
0 30 894 1306 2059 592 1474 236 1214 473 1171 275 186
31 215
0 1 68 117 72 73 985 2129 616 1944 69 160 131 1214 68 5
0 200 238 117 72 1628 429 1171 275 186 429 2932 368 181
0 228 2884 3118 302 604 2080 275 1432 225 81 5589 649 6
850 851 2934 1022
0 117 72 160 1048 545 68 1101 1122 689 186 89 87 305 82
184
0 545 68 50 12 1027 2352 32 6 449 87 160 2104 68 32 6
0 200 3029 689 1318 420 84 1707 213 545 68 3947 519 200
26 236 1419 599
0 200 238 1419 599 1083 1384 92 160 381 84 1878 1104 6
0 42 2593 1478 153 50 1048 381 1122 3681 3412 1419 599
0 200 117 72 2593 54 456 1531 275 501 146 1478 912 705
3 54 312 746 456
0 449 1188 599 545 84 381 686 1461 545 68 1183
0 1997 495 531 2898 297 2195 89 87 50
0 111 545 84 244 78 214 150 131 99 2932 368 63 3161 680
942 1104 84 2995 69 84 1461 1813 160 275 207 508 1174 4
963 2934 1445 616 1104
0 888 820 1813 887 50 1445 1083 627 6920 80
0 200 117 72 84 2188 1168 984 1202 649 14 820
0 1605 1164 7092 642 84
0 1 740 71 1581 161
0 117 72 86 591 135 120 146 63 238 95
0 578 428 669 529 119 669 302 5026 247 133 275 1474 330
2
0 200 335 6 16 117 1813 275 336 545 84 969
0 1 1086 117 228 604 275 428 81 280
0 1 117 213 2926 497 4371 73 335 6
0 64 50 5589 117 72 1942 280 81 275 335 6
0 886 6 484 751 68 247
0 1333 4 888 99 341 983 614 1474 131 1214 929 737 52 0
0 117 72 259 336 117 589 1270 135 120 146
0 117 1787 2593 54 135 5531 812 117
0 1209 501 117 72 234 609 117 186
0 1 64 531 929 2448 310 84 1707 678 428 3282 286 468 302
0 117 1826 778 336 1778 1171 275 186
```

article.features

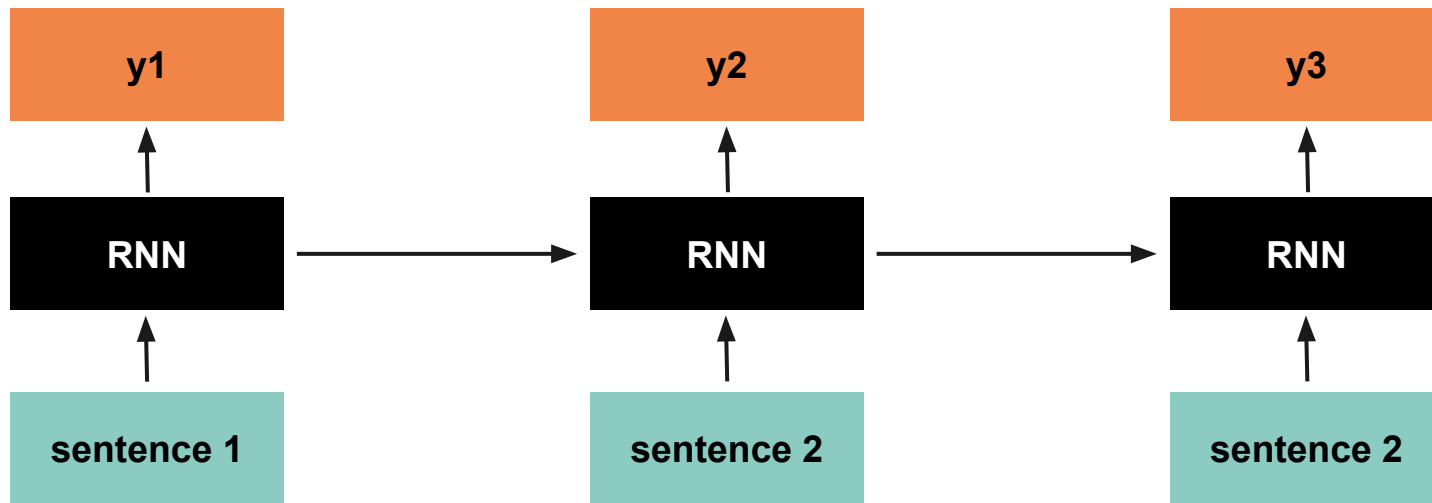


# Approach

- Extraction-based
- Single document



# Model Implementation





## ABSTRACT

Current work in surface realization concentrates on the use of general abstract algorithms that interpret large reversible **grammars**. Only little attention has been paid so far to the many small and simple applications that require coverage of a small sublanguage at different degrees of sophistication. The system **TG 2** described in this paper **can be smoothly integrated with deep generation processes integrates canned text templates and context free rules into a single formalism allows for both textual and tabular output** and it can be parameterized according to **linguistic** preferences. These features are based on suitably restricted production system techniques and on a generic backtracking regime.

## MODEL-GENERATED SUMMARY

Current work in surface realization concentrates on the use of general abstract algorithms that interpret declaratively defined non directional **grammars**. It is claimed that this way a grammar can be reused for parsing and generation or a generator can interpret different grammars eg in machine translation. Adapting their semantics layer to a generation algorithm and thus achieving reversibility can turn out to be a difficult enterprise. In particular **TG 2** **can be smoothly integrated with deep generation processes integrates canned text templates and context free rules into a single formalism allows for both textual and tabular output** efficiently reuses generated substrings for additional solutions and can be parameterized according to **linguistic** properties regarding style grammar fine grained rhetorics etc.



# Results

	Recall	Precision	F-Score
ROUGE-1	<b>0.04135</b>	<b>1</b>	<b>0.07941</b>
ROUGE-2	<b>0.04335</b>	<b>1</b>	<b>0.08309</b>
ROUGE-L	<b>0.08495</b>	<b>1</b>	<b>0.1566</b>



# Future Work

- More feature engineering
  - Word embeddings? Metadata?
- Hyperparameter tuning
- Utilize the full MEDLINE/PubMed dataset
- Baselines



**Questions?**