

AutoPropp

AutoPropp:

Toward the Automatic Markup, Classification, and
Annotation of Russian Magic Tales

Amicus Workshop

October 21, Vienna

Scott Alexander Malec

Software Engineer

Carnegie Mellon University

<http://www.maleclabs.com/Propp>

AutoPropp

- A work in progress, AutoPropp posits an algorithm to automatically assign the semantics of probable Proppian functions to candidate text segments in unannotated formulaic narratives (in this case, Russian magic tales), given training data that is marked up in PFTML.

AutoPropp

- Texts (scientific, folkloric, etc. (news – e.g., AP articles?)) are ordered hierarchies of content objects (OHCO hypothesis of Allen Renear (<http://www.stg.brown.edu/resources/stg/monographs/ohco.html>))
- Based on this assumption, one may posit means by which to tease out a tree hierarchy from a text itself.

AutoPropp

- How do we go about doing this?
- In this case, I will talk about what I hope is a novel context for Propp

Where -

- o Propp faces the "rigor" of computational linguistics
- o "bag of words models" - mixin' things up!

AutoPropp

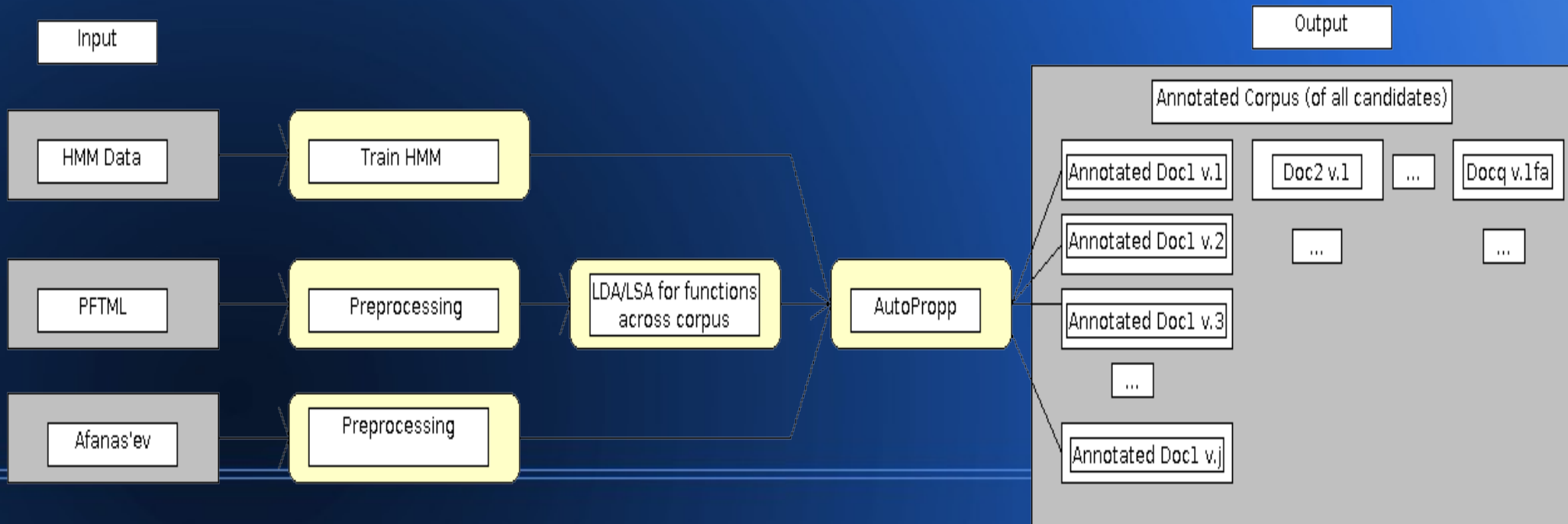
- AutoPropp is a way to deploy "new"(-ish) methods to grasp at the semantics of particular types of texts, e.g., formulaic (based on a pre-literary, formulaic oral transmission model (see Walter Ong) texts that possess a particular ordered hierarchy of context objects (OHCO) schema
- Hypothesis: by employing methods derived from supervised machine learning, one may automatically derive the structure of these content objects

AutoPropp: definition

- "AutoPropp: a work in progress, AutoPropp posits an algorithm to automatically assign the semantics of probable Proppian functions to candidate text segments in unannotated formulaic narratives (in this case, Russian magic tales), given training data that is marked up in PFTML."

AutoPropp

- How do I propose to do this?
- AutoPropp algorithm (in its "embryonic" stage) looks something like this diagram:



AutoPropp: the algorithm

- 1: Preprocess Raw Input and run HMM trainer.
- 2: Given PFTML corpus, create topic model of each function using LSA or LDA.
- 3: **for** each candidate passage s where $s \in$ Raw Input, **do**
 - Pick an s (given punctuation and other clues) and determine topics of candidate passage
 - Pick a hypothetical function f given HMM model of text
 - Determine similarity metric to topic given next function in HMM sequence
- 7: **end for**
- 8: Calculate for x permutations of s and functions
- 9: Pick MAX or top $y\%$ from MAX
- 10: Annotate texts from Raw Input in accordingly

AutoPropp

- Why do we care?
 - o We may want to develop (Turing test) machines that can automatically compelling narratives (generate narratives)
 - o For robots to become our intimate companions
 - o to identify and subsequently deceive political dissidents and mechanically crush them under the brutal heft of intellectual machinery that we have contracted out to the Police State for a pretty penny
- Because it is an interesting, ultimately intractable problem of which, regarding its structure, we might have the occasional faint lucid glimpse

AutoPropp

- Intuition: human beings understand narrative just like they understand and are able to grasp the rules of natural language in an inconceivable amount of time allowed by nature
- Therefore, language and, moreover, the sub-language of narrative itself, must be built into the human organism.

AutoPropp: the meat is DATA

- PFTML (or some variation thereof)
 - o Training corpus
 - o Extract topic model from each Proppian function (semantic unit in corpus)
 - Example:
 - o Donor function: "Baba Jaga gave Ivanko a golden apple"
- => develop training corpus

AutoPropp

- Markov models
 - transitive probability between semantic/Proppian functions/content objects
 - e.g., if D, then definitely an F
- => develop training corpus based on observations in order to optimize the automatic parsing of the (OHCO) text model
- Appendix III of Propp

AutoPropp

- Data
 - o English or Russian?
 - theoretically, it should not matter which language, given an effective stemmer, etc. – e.g., the curious case of Pavol Dobšinský (<http://zlatyfond.sme.sk/autor/40/Pavol-Dobsinsky>)
- Three rules of real estate: location, location, location
- Three rules of computational linguistics (or whatever this is): data, data, data

AutoPropp

- Create topicmodels for each Proppian function in training data from marked up PFTML

AutoPropp: relevant R packages

- XML
- tm
- openNLP
- lsa
- lda

AutoPropp

- XML

```
library(XML);
```

```
library(tm);
```

```
library(RCurl);
```

```
#url <- "http://clover.slavic.pitt.edu/sam/propp/have_a_little_byte/magicgeese.xml";
```

```
url <- "http://www.maleclabs.com/Propp/Corpus.xml";
```

```
tale <- xmlTreeParse(getURL(url), useInternal = T);
```

```
tale;
```

```
init <- xmlValue(getNodeSet(tale, "//Corpus//Folktale//Move//Preparation//InitialSituation")[[1]]);
```

```
init;
```

```
return <- xmlValue(getNodeSet(tale, "//Corpus//Folktale//Move//Return")[[1]]);
```

```
return;
```

AutoPropp

- tm

```
setwd("/home/propp/Desktop/Propp/AutoProppCorpus/PhilSources/Plato/");  
  
library("tm");  
  
txt <- system.file("texts", "txt", package = "tm");  
  
library(tm);  
  
library(Rgraphviz);  
  
afancorpus <- Corpus(DirSource(), readerControl = list(language = "eng"));  
  
afancorpus <- tm_map(afancorpus, removePunctuation);  
  
afancorpus <- tm_map(afancorpus, tolower);  
  
afancorpus <- tm_map(afancorpus, stripWhitespace);  
  
afancorpus <- tm_map(afancorpus, removeWords, stopwords("english"));  
  
afancorpus <- tm_map(afancorpus, stemDocument);  
  
afandtm <- DocumentTermMatrix(afancorpus);
```

AutoPropp

- openNLP

```
library(XML);
```

```
library(tm);
```

```
library(RCurl);
```

```
url <- "http://clover.slavic.pitt.edu/sam/propp/have_a_little_byte/magicgeese.xml";
```

```
#url <- "http://www.maleclabs.com/Propp/Corpus.xml";
```

```
tale <- xmlTreeParse(getURL(url), useInternal = T);
```

```
tale;
```

```
initsit <- xmlValue(getNodeSet(tale, "//Corpus//Folktale//Move//Preparation//InitialSituation")[[1]]);
```

```
initsit;
```

```
library(openNLP);
```

```
# sentDetect(initsit, language = "en", model = NULL);
```

```
y <- tagPOS(initsit, language = "en", model = NULL, tagdict = NULL);
```

```
y;
```

AutoPropp

- **Isa**

```
afan_Isa <- Isa(afandtm, dims = dimcalc_share(share = .5));
```

```
afan_Isa_k <- Isa(afandtm, dims = dimcalc_kaiser());
```

```
plot(afandtm);
```

```
inspect(afancorpus)[[20]];
```

```
summary(afan_Isa);
```

AutoPropp

- Ida

```
#LDA
```

```
p_LDA <- LDA(afandtm[1:250,], control = list(alpha = 0.1), k = 10);
```

```
post <- posterior(p_LDA, newdata = dtm[-c(1:250),]);
```

```
round(post$topics[1:5,], digits = 2);
```

```
get_terms(p_LDA, 5);
```

AutoPropp: problems to solve

- Problems to solve:
 - o size of the corpus
 - o consistency of the original markup
 - o infallible Propp
 - o NULL data
 - o noisy data
 - o learning and scoring