Open source NLP and machine learning for legal texts. What is Blackstone and how did we build it?

/AUGUST 9, 2019

BLACKSTONE

On 7 August 2019, the Incorporated Council of Law Reporting's newly formed research lab, ICLR&D, launched the prototype version of its legal natural language processing system, Blackstone.

For a more technical run down of what Blackstone is and how to use it, go over to the project's GitHub repository here. The purpose of this article is to provide a more general account of how Blackstone came into being and the thinking driving its development.

What is Blackstone?

Zoom out

Before diving into Blackstone, it's worth first zooming out and going back to the what led to ICLR&D's formation in the first place.

ICLR&D is the research division within the Incorporated Council of Law Reporting for England and Wales ("ICLR"). ICLR was itself founded in 1865 and is the creature of some genuinely innovative and disruptive Victorian thinking.

Prior to ICLR's establishment, the dissemination of case law in England and Wales was a chaotic and haphazard affair. Law reports and word-of-mouth were the only mechanisms available through which the existence and parameters of new precedents could be transmitted throughout the legal community. Word-of-mouth had obvious deficiencies and the law reports that were available were expensive, incomplete, slow to arrive and often inaccurate.

For a common law jurisdiction founded on the doctrine of precedent, the barriers to accurately disseminating the content of the law as decided by judges were becoming a real problem. To deal with what was a crisis waiting to happen for the administration of justice, senior members of the legal profession joined forces to devise a way forward so that the courts and practitioners had affordable and timely access to accurate reports of cases that changed the law.

The specification for the way forward was to establish a single body that would be responsible for monitoring the courts and producing reports of cases that needed reporting. There would be a clear and rigorous set of benchmarks that would be used to determine whether a case ought or ought not to be reported. There would be a single, well-structured style governing the presentation of the reports. And the authors of the reports would be skilled drafters who had been called to the English bar.

That specification led to the founding of the Incorporated Council of Law Reporting in 1865. 154 years later, here we are.

Zoom back in

Clearly, things have moved on a bit since the 1860s where law reporting in the UK is concerned, but not as much as you'd probably think. The most notable changes over the last 150 years relate to the quantity of case law made available over online subscription platforms and the fact that search has taken over as the method for case law retrieval (cumulative indexes and the library shelves have given way). But, in all over respects, the *content itself* has remained much the same as it was during the reign of Queen Victoria.

Uncontrolled environments

Another thing that has not changed over the last century and a half is the fact that the judgments themselves continue to be produced in uncontrolled environments. The evolution of the tooling of judgment drafting has gone from pen and paper, to the typewriter and then on to Microsoft Word.

In contrast, recent years have seen a surge in the development of contract drafting and review technologies designed to cast the business of contract generation and review into a *controlled environment*.

The virtue of controlling the environment in which legal content is generated lies in the fact that it enables the re-use of that content for other useful purposes. In the case of contracts and other legal documents of a transactional nature (such as lease agreements) the systemisation and standardisation of their construction makes it possible build knowledge bases and analytical tools that can be used by law firms to more effectively identify surfaces of risk, error and inefficiency.

Judgments, purely by virtue of what they are and what they contain, are rich with latent features that can be mined for all sorts of socially beneficial applications, such as:

- The development of better search tools to enable the general public to locate and understand case law
- Deeper analysis of the development of laws with respect to broader social trends and concerns
- Aiding drafters of legislation, particularly when common law concepts are being transposed into primary legislation
- Aiding law reporters to identify important, law-changing cases

It is unrealistic to expect that the business of judgments-drafting will migrate from Microsoft Word (i.e. an uncontrolled environment) to a controlled environment in the foreseeable future and it goes without saying that in the meantime we want the judges to continue focussing their efforts on ensuring that their judgments are correct in law, fully reasoned and given as quickly as possible.

Back to the original questions

It's easier, against that background, to articulate why ICLR&D came into being: ICLR, as a charitable organisation charged with providing systematic coverage of case law for a century and a half, recognised that technological innovation in the primary law space was significantly lagging behind the pace of technological innovation and change taking place in the commercial settings of large law firms. ICLR&D was therefore formed to provide ICLR with the resources it would need to begin carrying out more advanced research in the field of legal informatics to help us catch up.

So, what is Blackstone?

Blackstone is an experimental project to investigate the ways in natural language processing can be used to impose control and structure on legal content generated in uncontrolled environments. The project's deliverable is an open source piece of software, the Blackstone library, that allows researchers and engineers to automatically extract information from long, unstructured legal texts (such as judgments, skeleton arguments, scholarly articles, Law Commission reports, pleadings etc.)

Deep dive into Blackstone and the problem space

Blackstone is founded on the assumption that long and unstructured legal texts contain elements and characteristics that can be harnessed in a systematic way to improve our understanding of the meaning of the text and how it might fit into a larger corpus of text. Consider the following extract taken from the UK Supreme Court's decision in *R v Horncastle* [2009] UKSC 14; [2010] 2 AC 373:

> **6** The appellants submit that an affirmative answer must be given to this principal issue. In each case it is submitted that the trial judge should have refused to admit the statement on the ground that it was a decisive element in the case against the appellants. This the judge could have done, either by "reading down" the relevant provisions of the 2003 Act so as to preclude the admission of hearsay evidence in such circumstances or by excluding it under section 78 of the Police and Criminal Evidence Act 1984 ("PACE").

7 In so submitting the appellants rely on a line of Strasbourg cases, culminating in the decision of the Fourth Section of the European Court of Human Rights ("the Chamber"), delivered on 20 January 2009, in the cases of *Al-Khawaja and Tahery v United Kingdom* (2009) 49 EHRR 1. In each of those applications statements had been admitted in evidence at a criminal trial of a witness who was not called to give evidence. The Strasbourg court held that, in each case, the statement was "the sole or, at least, the decisive basis" for the applicant's conviction. The court reviewed its own jurisprudence and concluded that this established that the rights of each applicant under articles 6(1) and 6(3)(d) had not been respected. The court took as its starting point the following statement in *Lucà v Italy* (2001) 36 EHRR 807, para 40:

> "where a conviction is based solely or to a decisive degree on

> > depositions that have been made by a person whom the accused has had no opportunity to examine or to have examined, whether during the investigation or at the trial, the rights of the defence are restricted to an extent that is incompatible with the guarantees provided by article 6."
>
> I shall call the test of fairness that this statement appears to require "the sole or decisive rule".

In this short extract, we can identify a range of elements and characteristics that may be potentially useful to capture. For example,

- The first sentence of paragraph 6 contains references to a submission and an issue in the case.
- The second sentence of paragraph 6 contains another submission and identifies the ground for the submission
- The last sentence of paragraph 6 contains two references to legislation, a reference to a

provision and an abbreviated form of a statute name.

- In the first sentence of paragraph 7, the verb "rely" appears with reference to a line of case law, a court is referred to and a decision of that court (including the name and citation for that decision) is cited
- In the third sentence there is a summary of what a court decided
- In the fourth sentence reference is made to provisions in the European Convention on Human Rights.
- Reference is made to another case and that case's name and citation is given
- There is a block quote of a specific paragraph in a related case.
- The very last sentence in paragraph 7 refers to a legal test

Blackstone's objective is to develop a free and open source system for the extraction of this sort of information from unstructured legal texts. To frame the problem a different way: *we know that there is lots of useful information buried in legal texts, so how can be go about extracting it automatically?*

How Blackstone is designed to solve the extraction problem

There are two available strategies that can be used to solve this extraction problem.

Rules

The first strategy involves devising a set of explicit rules ahead of time that define the sorts of information we want to extract. There are several benefits to a rules-based approach. First, it eliminates the "blackbox" problem, because *we* are setting, and are in a position to inspect, the rules governing the extraction process. Second, computationally rules are relatively inexpensive.

The problem is that rules only work well when you can be sure that you can legislate for every possible permutation of the extraction targets. Rules work incredibly well where the raw data is predictable. Long, unstructured legal writing is not predictable. For example, take case citations. We could set out to identify all of the various shapes and sizes a citation may take and build a canonical list of rules that deal with those shapes and sizes. But what do we do if the author of the text being processed introduces a citation we haven't seen before? What if the author has made a mistake that our rules cannot mitigate because we could not foresee the mistake ahead of time?

Predictions

The second strategy, involves developing a system capable of predicting whether or not a feature in the text is of interest and then categorising that element. The objective is to train a statistical model to form a sufficiently accurate yet generalised view of the things we are interested in extracting. As with the rules-based approach outlined above, there are trade-offs here, too.

The overwhelming benefit of the prediction-based approach to the extraction problem is that an adequately trained model removes the need to legislate for every possible permutation of the information we are seeking to extract. There are disadvantages, however. The first is that we lose the ability to peer inside and see precisely why the model does what it does — the "blackbox" problem. The second problem is that we are never likely to train the model to achieve 100% accuracy — we need to make peace with the fact that our model will fail to extract something it ought to have extracted and our model will mislabel things. From time to time it might do things that a human will regard as completely stupid!

Compromise

In developing Blackstone, we have opted to pursue a blended strategy: the extraction task is fundamentally prediction-driven, with rules buttressing the model at various points along the way.

How Blackstone was built

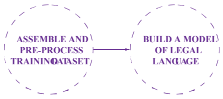
The development of Blackstone hinges on a single Python library called spaCy, which we selected because it's extremely well documented, robust, feature-rich and fast. What follows is a general overview of the process we followed the build Blackstone.

Training data

The starting point for the development of Blackstone was the assembly of the raw data that would be used to train the statistical models. Our data set was comprised of the following components:

- ICLR's archive of law reports, dating back to 1865
- ICLR's archive of unreported judgments, dating back to 2000

At this stage, the focus was to pre-process all of this data for use in training. Broadly speaking, this involved extracting each sentence from the dataset and removing things like leading, trailing and excessive spacing from the data.


ASSEMBLE AND PRE-PROCESS TRAINING DATASET
BUILD A MODEL OF LEGAL LANGUAGE

Build a model of legal language

Having assembled and cleaned up the data we would be using to train Blackstone's models we effectively had a very large text file that consisted of all of the sentences in all of the cases in the dataset, with one sentence per line. Whilst this in and of itself isn't that interesting, this giant list of sentences was bursting with information that could be used to give Blackstone a sense of the language used in case law and legal texts generally: we could use the giant list of sentences to train a *language model*.

Language modelling is an extremely complex and vibrant area of research, but a good way to think about it is that it's a way of training a machine to develop a sense of context for the words in our vocabulary (i.e. all of the unique words in the data we're using to train the language model on) . So for example, let's take the previous sentence as an example:

Language modelling is an ______ complex and vibrant area of research, but a good way to think _______ it is that it's a way of training a machine to develop a sense of context for the words in our vocabulary.

Two words have been removed from the sentence. The task is to infer what the missing words are based on our knowledge of the words in our vocabulary and our knowledge of the context in which certain words tend to appear.

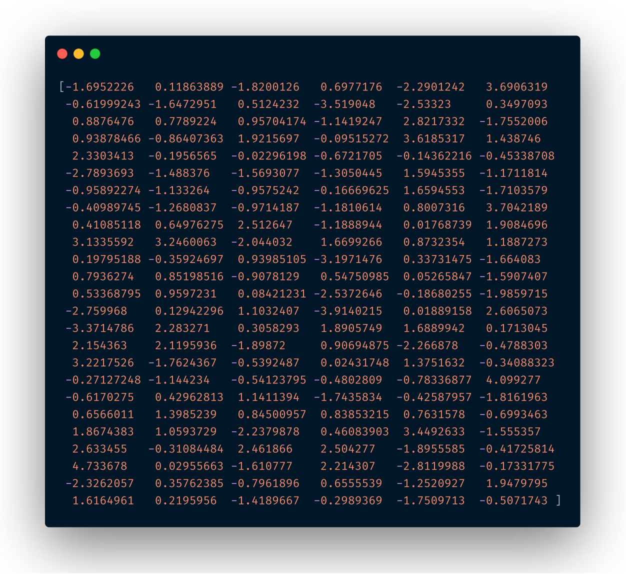
In the case of the first missing word we can infer from the surrounding context that we are probably looking for an adverb that qualifies the adjective "complex". So, acceptable possibilities are "extremely", "ludicrously", "highly" etc. We know it cannot be "very", because of presence of the determiner, "an".

In the case of the second missing word we can infer that it is likely to be a preposition that attaches to the "it" that immediately follows. The presence of "think" helps us narrow down the possibilities and there aren't many. "upon" and "on" are acceptable, yet awkward possibilities, but "about" is probably the best candidate.

So, in essence, this approach to language modelling turns on looking at each word in a sentence one at a time and scanning a certain number of words left and right of that word observation to build up a picture of its context.

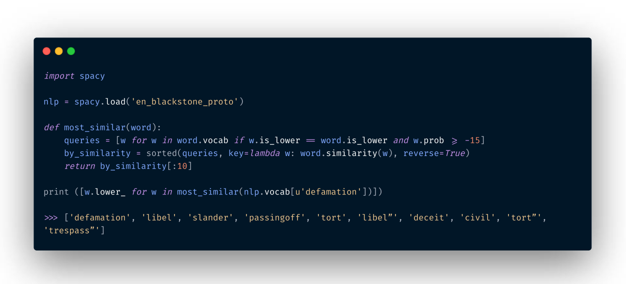
Blackstone's language model was trained using an algorithm called word2vec. What we get from word2vec is essentially a great big table in which each row is dedicated to each unique word in the giant list of sentences we built in the data assembly phase. Each column in the spreadsheet is a number, or a *vector*, which serves as the coordinates for each word in our little universe of words. Where things are liable to get a bit mind-bending is that these coordinates (the vectors) go well beyond the two-dimensional and three-dimensional planes we are used to reasoning in. Blackstone's language model consists of 374,302 unique words across 150 dimensions (which sounds big, but actually we'd like to make it much, much bigger!).

Here's what the vector for the word **defamation** looks like:



```
[-1.6952226   0.11863889 -1.8200126   0.6977176  -2.2901242   3.6906319
 -0.61999243 -1.6472951   0.5124232  -3.519048   -2.53323     0.3497093
  0.8876476   0.7789224   0.95704174 -1.1419247   2.8217332  -1.7552006
  0.93878466 -0.86407363  1.9215697  -0.09515272  3.6185317   1.438746
  2.3303413  -0.1956565  -0.02296198 -0.6721705  -0.14362216 -0.45338708
 -2.7893693  -1.488376   -1.5693077  -1.3050445   1.5945355  -1.1711814
 -0.95892274 -1.133264   -0.9575242  -0.16669625  1.6594553  -1.7103579
 -0.40989745 -1.2680837  -0.9714187  -1.1810614   0.8007316   3.7042189
  0.41085118  0.64976275  2.512647   -1.1888944   0.01768739  1.9084696
  3.1335592   3.2460063  -2.044032    1.6699266   0.8732354   1.1887273
  0.19795188 -0.35924697  0.93985105 -3.1971476   0.33731475 -1.664083
  0.7936274   0.85198516 -0.9078129   0.54750985  0.05265847 -1.5907407
  0.53368795  0.9597231   0.08421231 -2.5372646  -0.18680255 -1.9859715
 -2.759968    0.12942296  1.1032407  -3.9140215   0.01889158  2.6065073
 -3.3714786   2.283271    0.3058293   1.8905749   1.6889942   0.1713045
  2.154363    2.1195936  -1.89872     0.90694875 -2.266878   -0.4788303
  3.2217526  -1.7624367  -0.5392487   0.02431748  1.3751632  -0.34088323
 -0.27127248 -1.144234   -0.54123795 -0.4802809  -0.78336877  4.099277
 -0.6170275   0.42962813  1.1411394  -1.7435834  -0.42587957 -1.8161963
  0.6566011   1.3985239   0.84500957  0.83853215  0.7631578  -0.6993463
  1.8674383   1.0593729  -2.2379878   0.46083903  3.4492633  -1.555357
  2.633455   -0.31084484  2.461866    2.504277   -1.8955585  -0.41725814
  4.733678    0.02955663 -1.610777    2.214307   -2.8119988  -0.17331775
 -2.3262057   0.35762385 -0.7961896   0.6555539  -1.2520927   1.9479795
  1.6164961   0.2195956  -1.4189667  -0.2989369  -1.7509713  -0.5071743 ]
```

Whilst this numbers are pretty meaningless to us mere mortals, they provide the model with a really rich mathematical sense of the meaning of the words in the vocabulary and allows it to cluster similar concepts together. For example, we can ask Blackstone to show us the terms it has clustered in and around “defamation”:



```
import spacy

nlp = spacy.load('en_blackstone_proto')

def most_similar(word):
    queries = [w for w in word.vocab if w.is_lower == word.is_lower and w.prob >= -15]
    by_similarity = sorted(queries, key=lambda w: word.similarity(w), reverse=True)
    return by_similarity[:10]

print ([w.lower_ for w in most_similar(nlp.vocab[u'defamation'])])

>>> ['defamation', 'libel', 'slander', 'passingoff', 'tort', 'libel”', 'deceit', 'civil', 'tort”',
'trespass”']
```

We can see from the output that Blackstone’s language model has both successfully aligned “defamation” with “libel” and “slander” (themselves species of defamation) *and* acquired a sense of torts, which is pleasing because defamation is a tort!

ASSEMBLE AND PRE-PROCESS TRAINING DATASET
BUILD A MODEL OF LEGAL LANGUAGE
CONSTRUCT CORE PIPELINE

Construct the model’s core pipeline

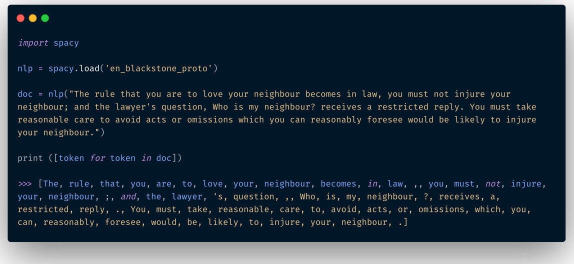
To recap, we have so far collected training data in the form of single sentences derived from ICLR’s archive of law reports and trained Blackstone’s base language model. The next step is to plugin three components that will be essential for the extraction tasks Blackstone was built to carry out.

This is where spaCy comes into its own. The three essential ingredients we add here are as follows.

The tokeniser

The tokeniser is the component in the model responsible for breaking down texts that are given to the model into segments. This is a crucial first step in pretty much every natural language processing pipeline.

We’ll apply the tokeniser to a small section of Lord Atkin’s seminal judgment in *Donoghue v Stevenson* [1932] AC 562:



```
import spacy

nlp = spacy.load('en_blackstone_proto')

doc = nlp("The rule that you are to love your neighbour becomes in law, you must not injure your
neighbour; and the lawyer's question, Who is my neighbour? receives a restricted reply. You must take
reasonable care to avoid acts or omissions which you can reasonably foresee would be likely to injure
your neighbour.")

print ([token for token in doc])

>>> [The, rule, that, you, are, to, love, your, neighbour, becomes, in, law, ,, you, must, not, injure,
your, neighbour, ;, and, the, lawyer, 's, question, ,, Who, is, my, neighbour, ?, receives, a,
restricted, reply, ., You, must, take, reasonable, care, to, avoid, acts, or, omissions, which, you,
can, reasonably, foresee, would, be, likely, to, injure, your, neighbour, .]
```

You'll see that each word token and punctuation mark has been separately accounted for, including the "'s" on "lawyer's". Not particularly interesting, but fundamental nevertheless.

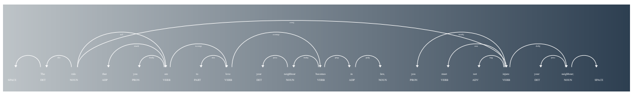
We'll consider the remaining two ingredients together.

Part-of-Speech Tagger and Dependency Parser

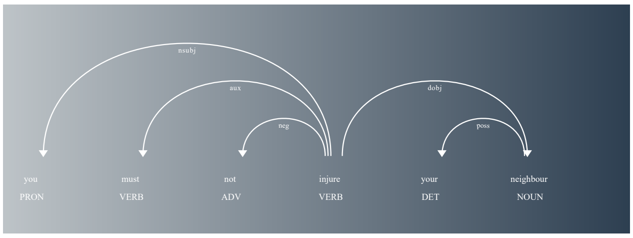
Part-of-speech tagging refers to the process of assigning a label to each token (see above) in a text according to its function in the sentence, i.e. is the token a noun, verb, adjective, punctuate mark, etc. Dependency parsing refers to the process of building a tree of the relationships between the tokens in the text. In combination, part-of-speech tagging and dependency parsing provide a wealth of insight into the semantic structure of a text.

The following images demonstrate part-of-speech tagging and dependency parsing.

This image takes the first clause of the extract from Lord Atkin's holding in *Donoghue* and shows the result of both the part-of-speech tagging and dependency parsing process. The capitalised labels at the bottom of the image denote the part-of-speech tag assigned to each word or punctuation token in the text and the directed arcs demonstrate the relationships between a *head* token and its dependents.

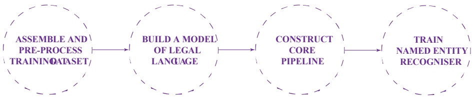


It is clearer when we focus in on a smaller part of text. The dependency arcs demonstrate the critical components in this short clause. The token “you” is being assigned as the nominal subject of “injure”, which makes sense because in this context *you* are being instructed that *you must not injure*. The token ”neighbour” is assigned as the direct object of “injure”, which again makes sense because the injuring being prohibited applies to the neighbour.



Part-of-speech tagging and dependency parsing are pretty dense areas of research. But for present purposes it helps to know that these two components play an important role in other functionality offered in the Blackstone model.

Before leaving this topic, it’s worth pointing out that Blackstone uses the tagger and parser from spaCy’s small English model. The reason for this is that as things currently stand there are no POS/Dependency datasets specifically trained on legal text. spaCy's tagger and parser actually work really well, but we would like to revisit this at some point in the future (if you are a computational linguist interested in building such a dataset, please contact us at research@iclr.co.uk).


ASSEMBLE AND PRE-PROCESS TRAINING DATASET
BUILD A MODEL OF LEGAL LANGUAGE
CONSTRUCT CORE PIPELINE
TRAIN NAMED ENTITY RECOGNISER

Train the Named Entity Recogniser

Again, let's pause to recap our progress at this point. So far we have:

- Assembled and processed our dataset
- Trained a language model on the dataset
- Constructed the core components on the natural language processing pipeline (the tokeniser, the part-of-speech tagger and the dependency parser).

We can now move on to the fun bit — training the named entity recogniser.

The named entity recogniser (NER) is responsible for the bulk of the heavy lifting required by Blackstone to extract information from legal text. Think of the NER as a model for spotting “things” we are interested in extracting from our text.

The prototype Blackstone model has been trained to recognise the following types of things (entities):

- Case names, e.g *Donoghue v Stevenson*
- Case citations, e.g. [1932] AC 562
- Instruments, e.g. Criminal Justice Act 2003
- Provisions, e.g. section 1(1)
- Courts, e.g. Court of Appeal
- Judges, e.g. Eady J or Lord Bingham

Our goal here was to train Blackstone to form an accurate yet *generalised* view of what these entities tend to look like and the contexts in which they tend to appear. At the development stage, there’s no way of know what users are going to throw at the model further down the line, so this ability for the model to generalise and say ”aha, this looks a bit like a citation to me and judging by it’s context it is probably is a citation” is very important, because it will allow the model to detect citations it didn’t encounter during training (for context, there are 728 citation structures beginning with the letter ”R”).

How the NER was trained

Blackstone's NER model is a *supervised* model, which means we taught it to recognise the entity types we were interested in capturing by showing it lots of examples of those entity types.

Remember our giant list of sentences from the beginning? We needed that again here.

The first job was to generate the examples we'd use for training. The training data needed to be in a specific format, that looks like this:

```
('Mr Ward argues that the width of
that power of recall is not cut
down by the principle set out in R
(B) v Ashworth Hospital Authority
[2003] 1 WLR 1886, which is
concerned with the situation of a
patient while he is in hospital.',
{'entities':[(98, 133,
'CASENAME'), (134, 151,
'CITATION')]})
```

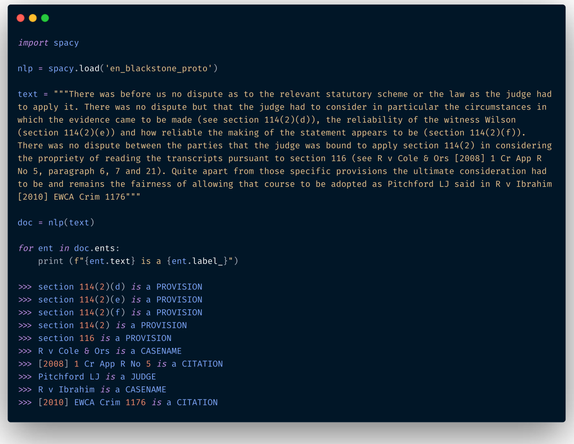
The block of text above contains two components that turn it into a sample our model can learn from. The first is the text of the sentence ("Mr Ward argues…"). The second component, which has been added, is a list of the things in that sentence we want the model to learn to recognise. A case is cited, *R (B) v Ashworth Hospital Authority* [2003] 1 WLR 1886, There are therefore two entities in this sentence we want to show the model during training: the name of the case (*R (B) v Ashworth Hospital Authority*) and the its citation ([2003] 1 WLR 1886).

The purple section identifies the type of entities present in the sentence together with their location in the sentence. So, for example from the 98th character to the 133rd character in the sentence, we have a case name. And, from the 134th character to the 151st character, we have a citation.

We processed all of the sentences in our giant list of sentences to generate a separate giant list of entries that looked like the one above, including examples of the other four entity types the prototype model Blackstone detects.

Before we could start training the NER model we needed to take one final and critical step. We took the giant list of entries we just generated and grabbed 30 percent of those entries and stored them separately from the remaining 70 percent. The remaining 70 percent would be used during training as examples of the sorts of things we needed Blackstone to predict. The 30 percent would be kept well away from the process and would be used to test how well Blackstone was doing at learning from the examples we were giving it during training (we obviously didn't want to give the model the sight of the data we'd be using to test it because that's just like allowing it to flick to the back of the book to find the answer).

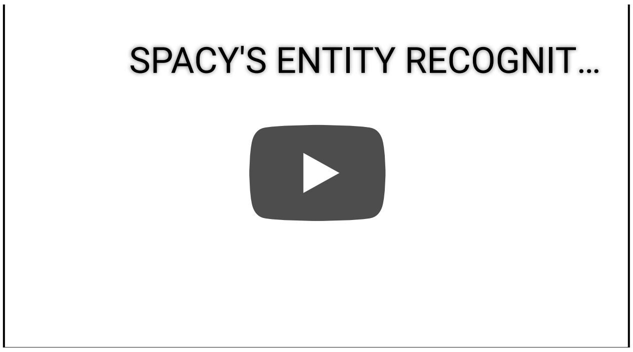
Here's an example of Blackstone's NER model being applied to para 13 of the Court of Appeal's judgment in *R v Davis (Iain)* [2008] EWCA Crim 1735:

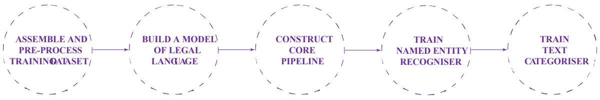


```
import spacy

nlp = spacy.load('en_blackstone_proto')

text = """There was before us no dispute as to the relevant statutory scheme or the law as the judge had
to apply it. There was no dispute but that the judge had to consider in particular the circumstances in
which the evidence came to be made (see section 114(2)(d)), the reliability of the witness Wilson
(section 114(2)(e)) and how reliable the making of the statement appears to be (section 114(2)(f)).
There was no dispute between the parties that the judge was bound to apply section 114(2) in considering
the propriety of reading the transcripts pursuant to section 116 (see R v Cole & Ors [2008] 1 Cr App R
No 5, paragraph 6, 7 and 21). Quite apart from those specific provisions the ultimate consideration had
to be and remains the fairness of allowing that course to be adopted as Pitchford LJ said in R v Ibrahim
[2010] EWCA Crim 1176"""

doc = nlp(text)

for ent in doc.ents:
    print (f"{ent.text} is a {ent.label_}")

>>> section 114(2)(d) is a PROVISION
>>> section 114(2)(e) is a PROVISION
>>> section 114(2)(f) is a PROVISION
>>> section 114(2) is a PROVISION
>>> section 116 is a PROVISION
>>> R v Cole & Ors is a CASENAME
>>> [2008] 1 Cr App R No 5 is a CITATION
>>> Pitchford LJ is a JUDGE
>>> R v Ibrahim is a CASENAME
>>> [2010] EWCA Crim 1176 is a CITATION
```

The current state of the NER in the prototype Blackstone model is not perfect (and, just by virtue of how these things work, it likely never will be). But all in all, as the example above shows, the model performs pretty well.

As with the rest of the components in the Blackstone model, named entity recognition is a pretty tricky area of research. But, if you're interested in learning more about how this works under the hood, we'd encourage you to watch the following video by spaCy's author, Matthew Honnibal.




ASSEMBLE AND PRE-PROCESS TRAINING DATASET
BUILD A MODEL OF LEGAL LANGUAGE
CONSTRUCT CORE PIPELINE
TRAIN NAMED ENTITY RECOGNISER
TRAIN TEXT CATEGORISER

Train the text categoriser

Initially, we planned on having a larger selection of target entities in Blackstone's NER model that could also detect instances in which things like a conclusion was being expressed or a legal issue was being flagged. In truth, it wasn't until relatively recently that we learned that this approach was flawed, because these additional targets of interest were not really named entities (many thanks to Mark Neumann of AI2 for the guidance here.)

This last minute correction in course led to the development of the final component built into the prototype Blackstone model; the text categoriser.

The aim here was train Blackstone to determine whether or not a given sentence could be categorised according to one of five category types:

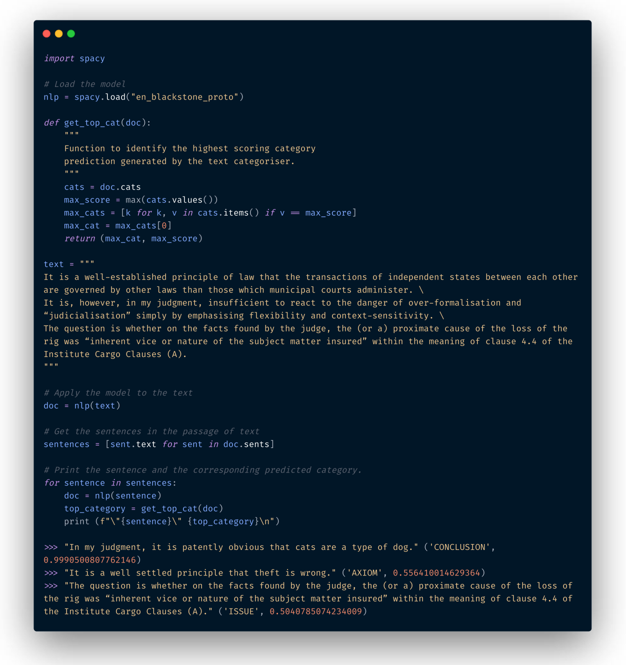
- Axiom — the sentence sets out an established or well settled principle of law (inspired by the Stanford Codex project, wellsettled).
- Conclusion — the sentence expresses a conclusion or some other determination
- Issue — the sentence discusses an issue that is relevant in the context of the broader document
- Legal Test — the sentence discusses some sort of legal test
- Uncategorised — the sentence does not fall into any of the previous four categories

The training strategy for this component of Blackstone was not all that different to the strategy used to train the NER. We'd return to our original giant list of sentences and build a labelled training set.

Here's an example of a sample used to train the text categoriser (in this case, the example is a conclusion):

```
('If no reasonable person would
contemplate “fencing” or providing
a “fixed guard” to a piece of
machinery it is not, in my
judgment, a part of machinery or
machinery as that word is used in
the section.', {"cats":
{"CONCLUSION": 1, "AXIOM": 0,
"ISSUE": 0, "LEGAL_TEST": 0,
"UNCAT": 0}})
```

And here is Blackstone's text categoriser in action:



```
import spacy

# Load the model
nlp = spacy.load("en_blackstone_proto")

def get_top_cat(doc):
    """
    Function to identify the highest scoring category
    prediction generated by the text categoriser.
    """
    cats = doc.cats
    max_score = max(cats.values())
    max_cats = [k for k, v in cats.items() if v == max_score]
    max_cat = max_cats[0]
    return (max_cat, max_score)

text = """
It is a well-established principle of law that the transactions of independent states between each other
are governed by other laws than those which municipal courts administer. \
It is, however, in my judgment, insufficient to react to the danger of over-formalisation and
“judicialisation” simply by emphasising flexibility and context-sensitivity. \
The question is whether on the facts found by the judge, the (or a) proximate cause of the loss of the
rig was “inherent vice or nature of the subject matter insured” within the meaning of clause 4.4 of the
Institute Cargo Clauses (A).
"""

# Apply the model to the text
doc = nlp(text)

# Get the sentences in the passage of text
sentences = [sent.text for sent in doc.sents]

# Print the sentence and the corresponding predicted category.
for sentence in sentences:
    doc = nlp(sentence)
    top_category = get_top_cat(doc)
    print (f"\"{sentence}\" {top_category}\n")

>>> "In my judgment, it is patently obvious that cats are a type of dog." ('CONCLUSION',
0.9990500807762146)
>>> "It is a well settled principle that theft is wrong." ('AXIOM', 0.556410014629364)
>>> "The question is whether on the facts found by the judge, the (or a) proximate cause of the loss of
the rig was “inherent vice or nature of the subject matter insured” within the meaning of clause 4.4 of
the Institute Cargo Clauses (A)." ('ISSUE', 0.5040785074234009)
```

Where to from here?

Now that the prototype model is openly available on GitHub, the immediate next steps are to return to the training phase to tighten Blackstone's predictions up a bit. Needless to say, these are the very early stages in a far longer programme of NLP research at ICLR&D. Either way, we think this project is a solid opening salvo in our bid to move open research and development with law and technology into centre stage in the UK.

Posted in Data science, Blackstone Tags: NLP, Natural language processing, Open source, Open law, Machine learning, Deep learning, spaCy

3 Likes / Share

Comments (0)

Newest First Subscribe via e-mail

Preview POST COMMENT…

← Blackstone Conc…

Our toolbox →

web: www.iclr.co.uk email: research@iclr.co.uk

Charity No.250605 Limited by Guarantee - Company Registered No.5034