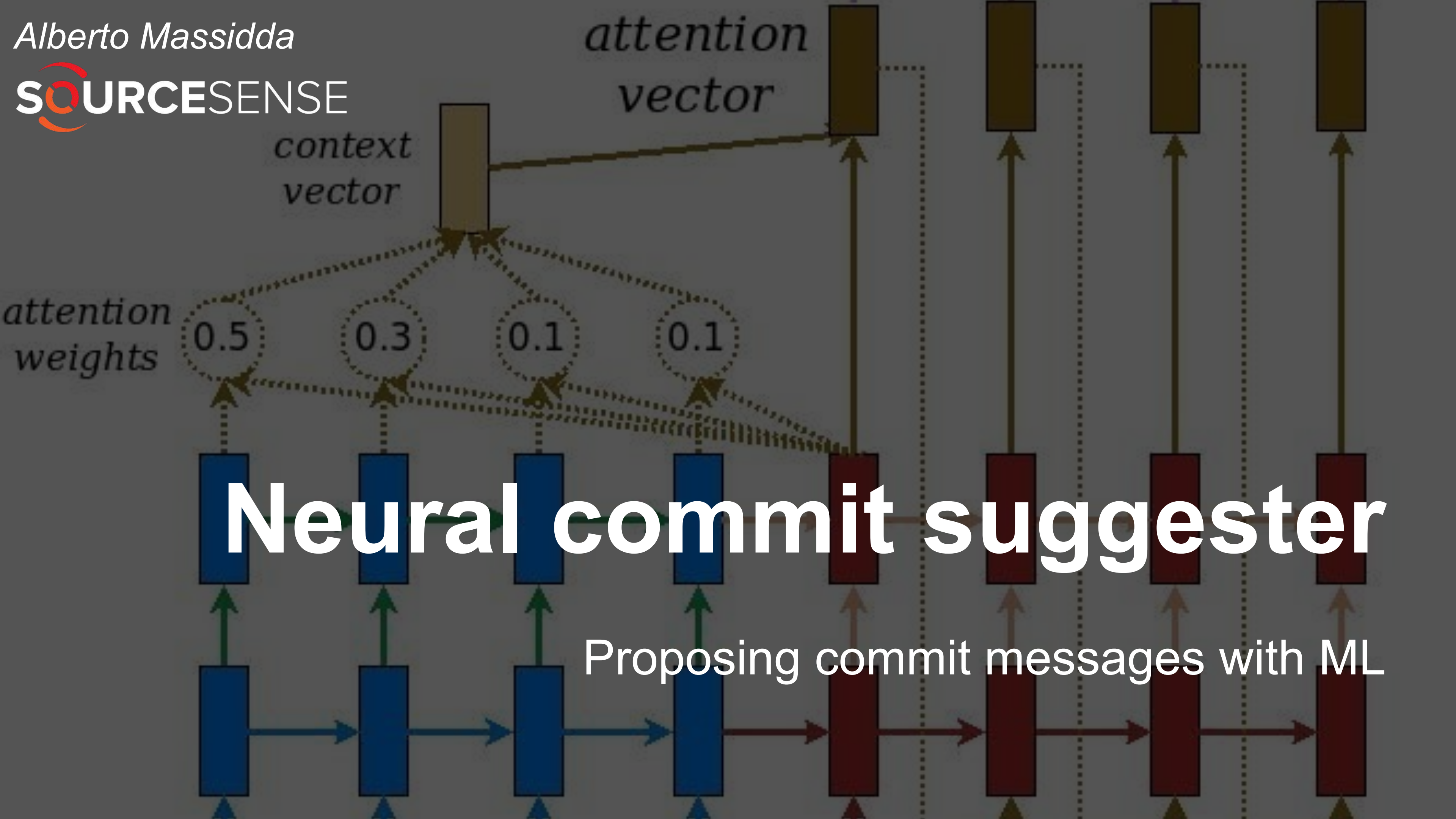
Alberto Massidda
SOURCESENSE



Neural commit suggester

Proposing commit messages with ML

Who we are

- Founded in 2001;
- Branches in Milan, Rome and London;
- Market leader in enterprise ready solutions based on Open Source tech;
- Expertise:
  - DevOps
  - Cloud
  - ML
  - BigData and many more...

Motivation for a commit suggester

We could:

- just help the developer in picking a nice message (aid suggestion);
- catch bad commit messages too far from suggestion (gate suggestion);
    - *Jenkins rejects the pull request due to lousy commit message!*

We don’t want/need:

- messages based on templates;
- messages that summarize *what* changed and not *why*;

SOURCESENSE

Generation as summarization

Generalize what was the intent of the coder, at least at a low level.

A change of code always comes with a commit message, describing the full change.

In essence, generating a commit message is generating a summary of the changes.

Generation as summarization

Diff patches provide a very focused source of “code-to-summary” mapping.

```
--- a/kubernetes/ansible/ansible_config/tasks/docker.yml
+++ b/kubernetes/ansible/ansible_config/tasks/docker.yml
@@ -1,5 +1,8 @@
 - name: Create docker default nexus auth
   template:
     src: ../../ansible/roles/docker/files/docker-config_staging.json.j2
-    dest: ../../ansible/roles/docker/files/docker-config_staging.json
+    dest: "{{item}}"
     force: true
+  with_items:
+    - ../../ansible/roles/jenkins/files/docker-config.json
+    - ../../ansible/roles/docker/files/docker-config_staging.json
```

Neural Machine Translation to the rescue

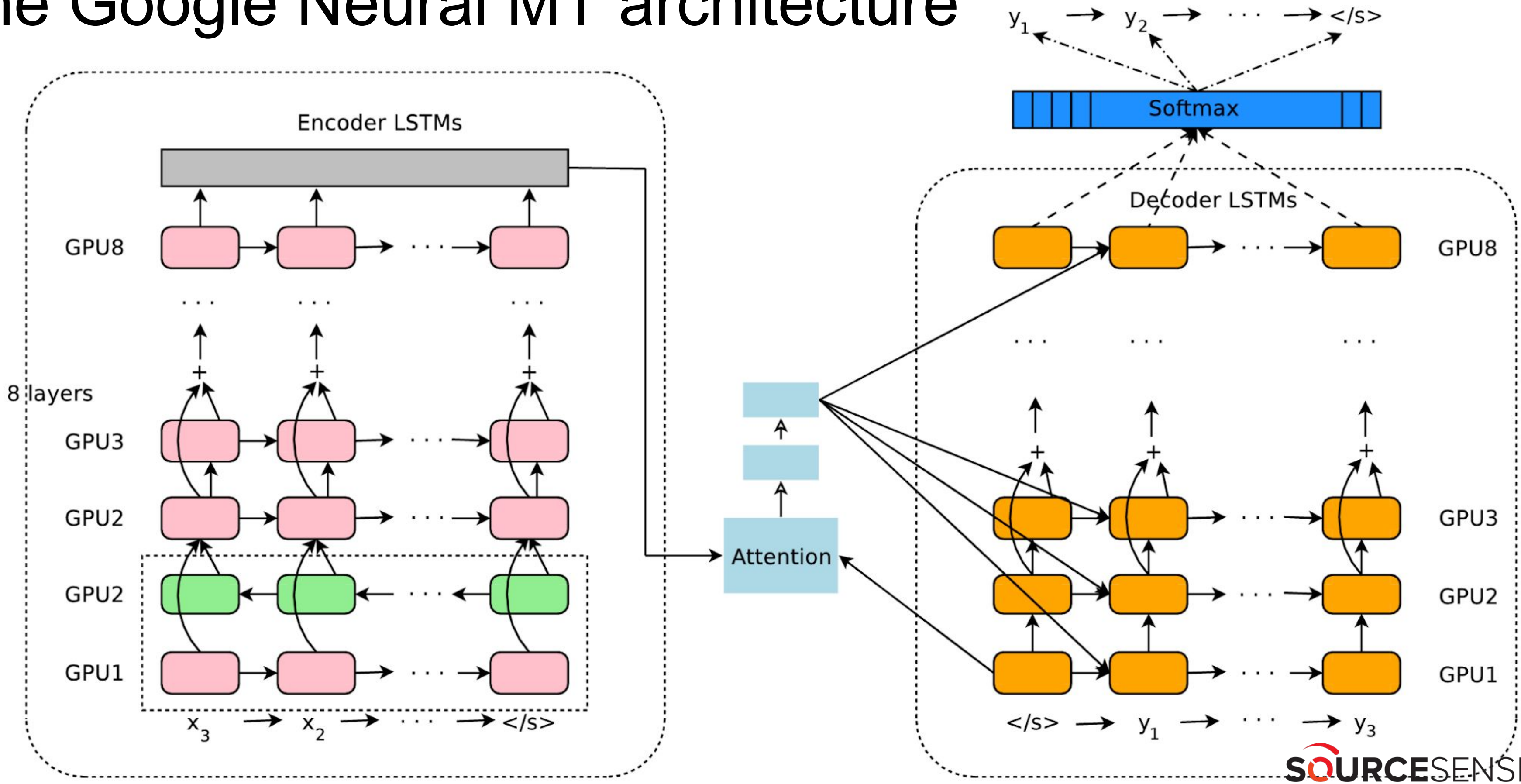
We need a way to learn mapping from diffs to a natural language summary.

Machine Translation can help!

The whole point of statistical (and later, neural) machine translation is to infer a mapping between languages, by means of co-occurrences counting or vector embedding manipulations.

We need an architecture and a dataset.

The Google Neural MT architecture



Dataset

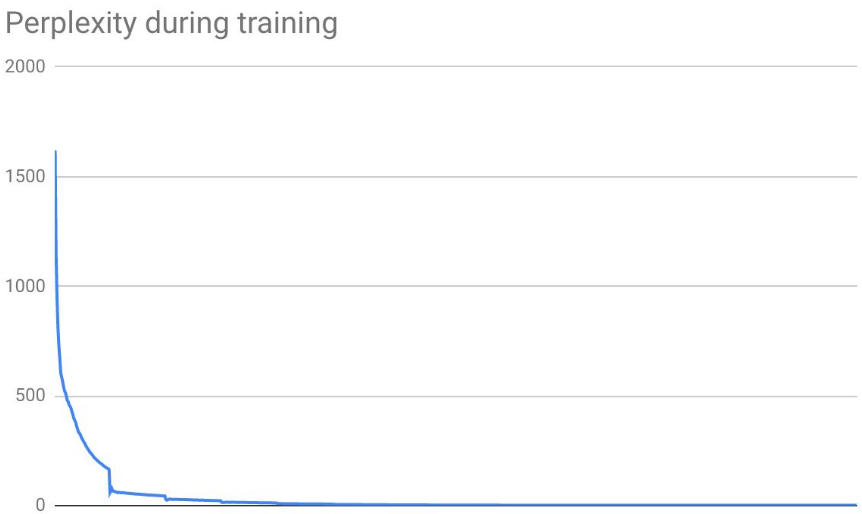
- We used the commit data set provided by *Jiang and McMillan*
    - **2M commits top 1000 Java projects on GitHub.**
- Extract first sentence only.
- Only diff patch, no issuer, no commit hash.
- Tokenization for white space, keep camel casing and punctuation.
- No merge/rollback. No diffs > 1MB.
    - **1.8M commits left**
- Source token length: 100 max. Target token length: 30 max.
    - **75k commits left**
- "Verb - Direct Object" only messages (filtered via CoreNLP POS tagging)
    - **32k commits left**
    - **3k testing, 3k validation, the rest 26k for training**

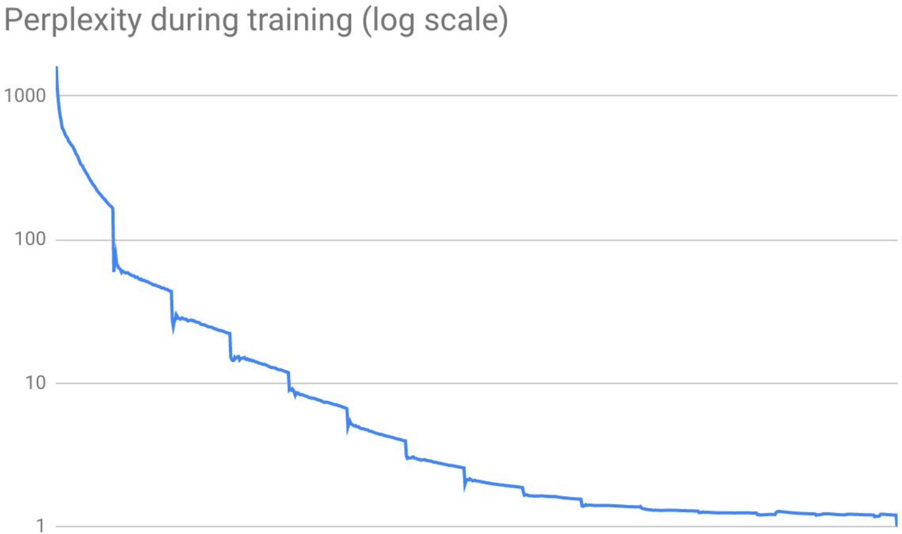
SOURCESENSE

Train time

We used *Sockeye*, a seq2seq framework based on AWS MXNet.

Traning happened on a *p2.xlarge* (Tesla K80) and a *p3.2xlarge* (Tesla V100).





SOURCESENSE

Results 5 hours (242 epochs, 43k minibatch) later

```
--- a/src/main/groovy/util/ConfigObject.java
+++ b/src/main/groovy/util/ConfigObject.java
* /
package groovy.util;
- import groovy.lang.Closure;
- import groovy.lang.GroovyObject;
import groovy.lang.GroovyObjectSupport;
import groovy.lang.Writable;
import org.codehaus.groovy.runtime.DefaultGroovyMethods
```

Human: *Removed non-needed imports*
Machine: *Remove unused import*

SOURCESENSE

Results 5 hours (242 epochs, 43k minibatch) later

```
--- a / python / README
+++ b / python / README
Python - to - libsvm interface
+ Table of Contents
+ = = = = = = = = = = = = = = = = = =
+
+ - Introduction
+ - Installation
+ - Usage
+ - Examples
+
Introduction
= = = = = = = = = = = =
```

Human: *add table of contents in python / README*
Machine: *add table of contents in python / README*

SOURCESENSE

Results 5 hours (242 epochs, 43k minibatch) later

```
--- a / build . gradle
+++ b / build . gradle
buildscript {
  jcenter ( )
}
dependencies {
  - classpath ' com . android . tools . build : gradle : 2 . 2 . 0 '
  + classpath ' com . android . tools . build : gradle : 2 . 2 . 2 '
}
}
```

Human: *update gradle*
Machine: *Updated build tools version*

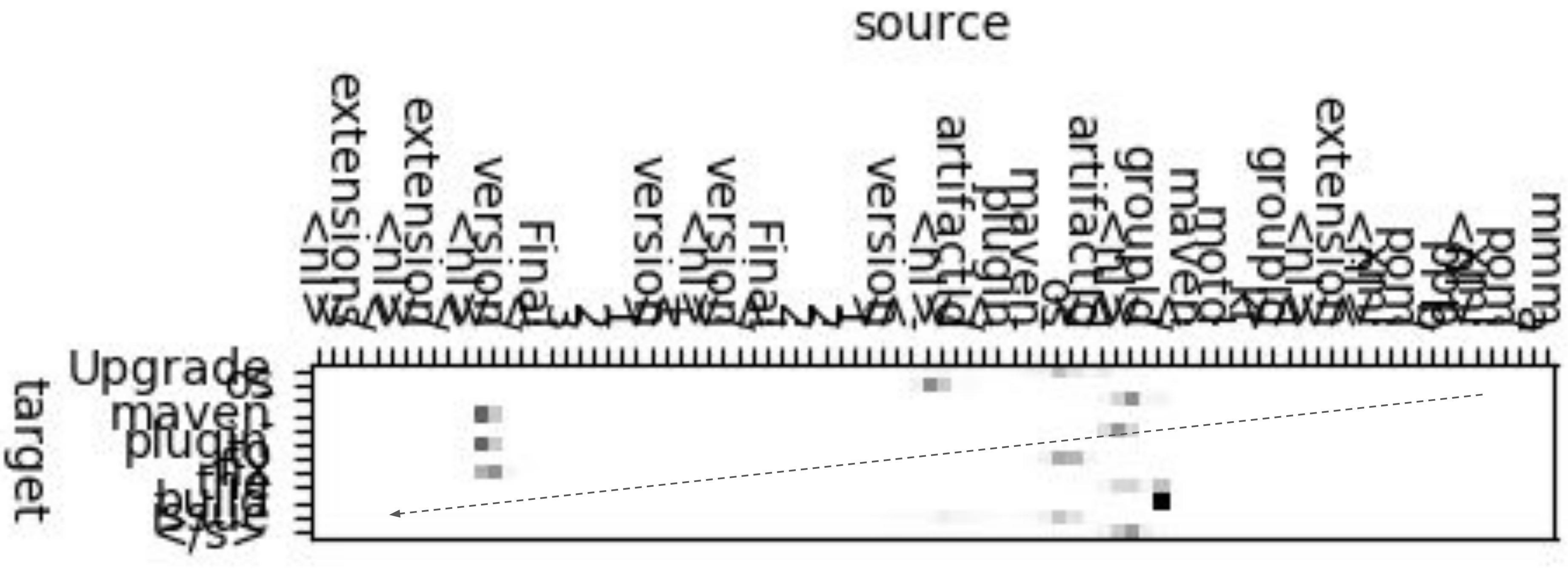
Results 5 hours (242 epochs, 43k minibatch) later

```
--- a / pom . xml
+++ b / pom . xml
< extension >
< groupId > kr . motd . maven < / groupId >
< artifactId > os - maven - plugin < / artifactId >
- < version > 1 . 2 . 2 . Final < / version >
+ < version > 1 . 2 . 3 . Final < / version >
< / extension >
< / extensions >
```

Human: *Upgrade os - maven - plugin to fix an issue with IntelliJ IDEA on Windows*
Machine: *Upgrade os - maven - plugin to fix the build issue*

SOURCESENSE

Attention model plot



Profit? Well…

BLEU score 37.6

CHRF: 40.5

The model has learned:

★ fluent English;
★ very interesting correlations in short commit patches.

SOURCESENSE

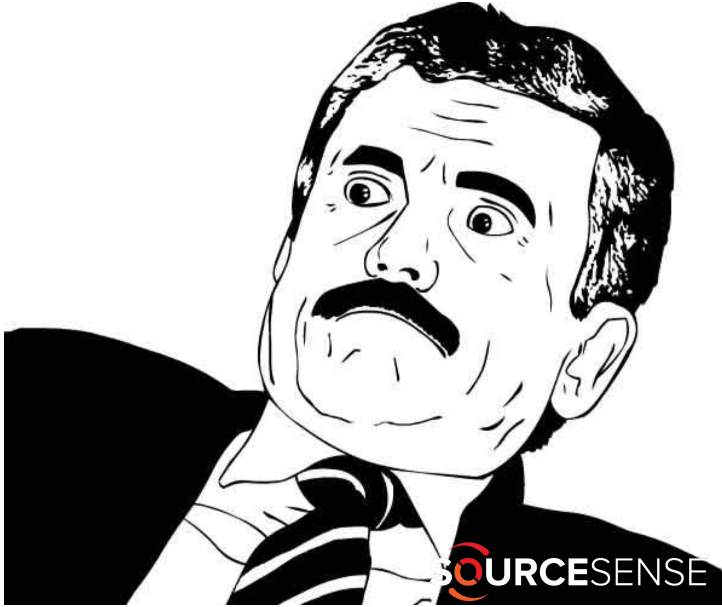
Profit? Well… No.

But, overall, **the error rate for long patches is embarrassing**:
a LOT of sentences are totally incoherent with diffs patches.
That’s why the dataset is so picked.

Example (and I have piles of this):

Human: *Change default fbo cache size to 0*

Machine: *Add unused import for NOPASS .*



A nice thing about software technologies

You learn the most out of them
by watching them **fail**

SOURCESENSE

Extremely difficult task in practice

Vanilla MT architecture not optimized for task.

- **Length imbalance**: input sentences 2-10x longer than output.
- **Decoder RNN is fluent**: output within 10 tokens on average.
- **Poor context performance**: due to encoder RNN length, difficult for LSTM to remember 500 words context. Sentence complexity affects negatively Attention model, who can't keep up with such a big and sparse state.
- **Memory problems**: GNMT trains well, Transformer goes OOM immediately.

SOURCESENSE

A better architecture proposal: HAN-NMT

The main source of chaos stems from the input length and complexity:
we cram together **insertions**, **ablations** and **context**.

It would make much more sense to adopt a multi-encoder network:

- 1 encoder for insertions;
- 1 encoder for ablations;
- 1 encoder for context;
- **Hierarchical Attention Network to rule out uninfluent encoders**;
- 1 decoder for the output.

Much in the spirit of Transformer multi-headed attention.

SOURCESENSE

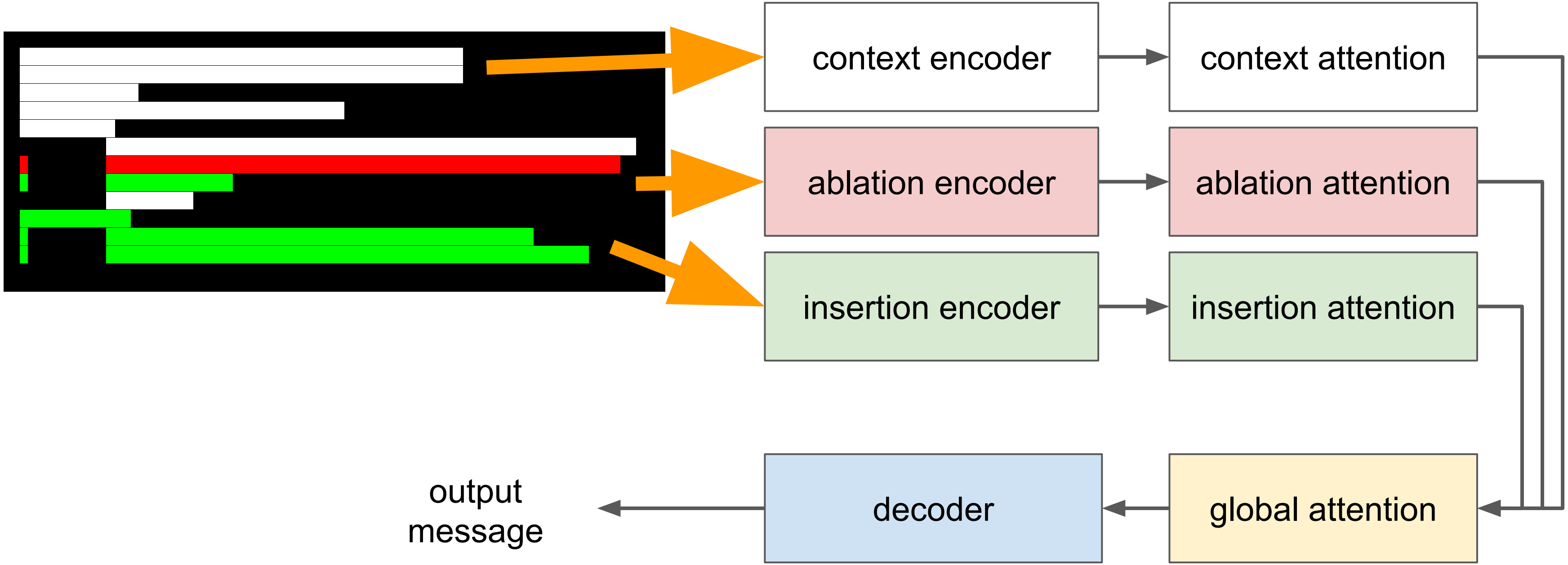
Remember this?

Diff patch provides a natural way to separate contexts.

```
--- a/kubernetes/ansible/ansible_config/tasks/docker.yml
+++ b/kubernetes/ansible/ansible_config/tasks/docker.yml
@@ -1,5 +1,8 @@
 - name: Create docker default nexus auth
   template:
     src: ../../ansible/roles/docker/files/docker-config_staging.json.j2
-    dest: ../../ansible/roles/docker/files/docker-config_staging.json
+    dest: "{{item}}"
     force: true
+  with_items:
+    - ../../ansible/roles/jenkins/files/docker-config.json
+    - ../../ansible/roles/docker/files/docker-config_staging.json
```

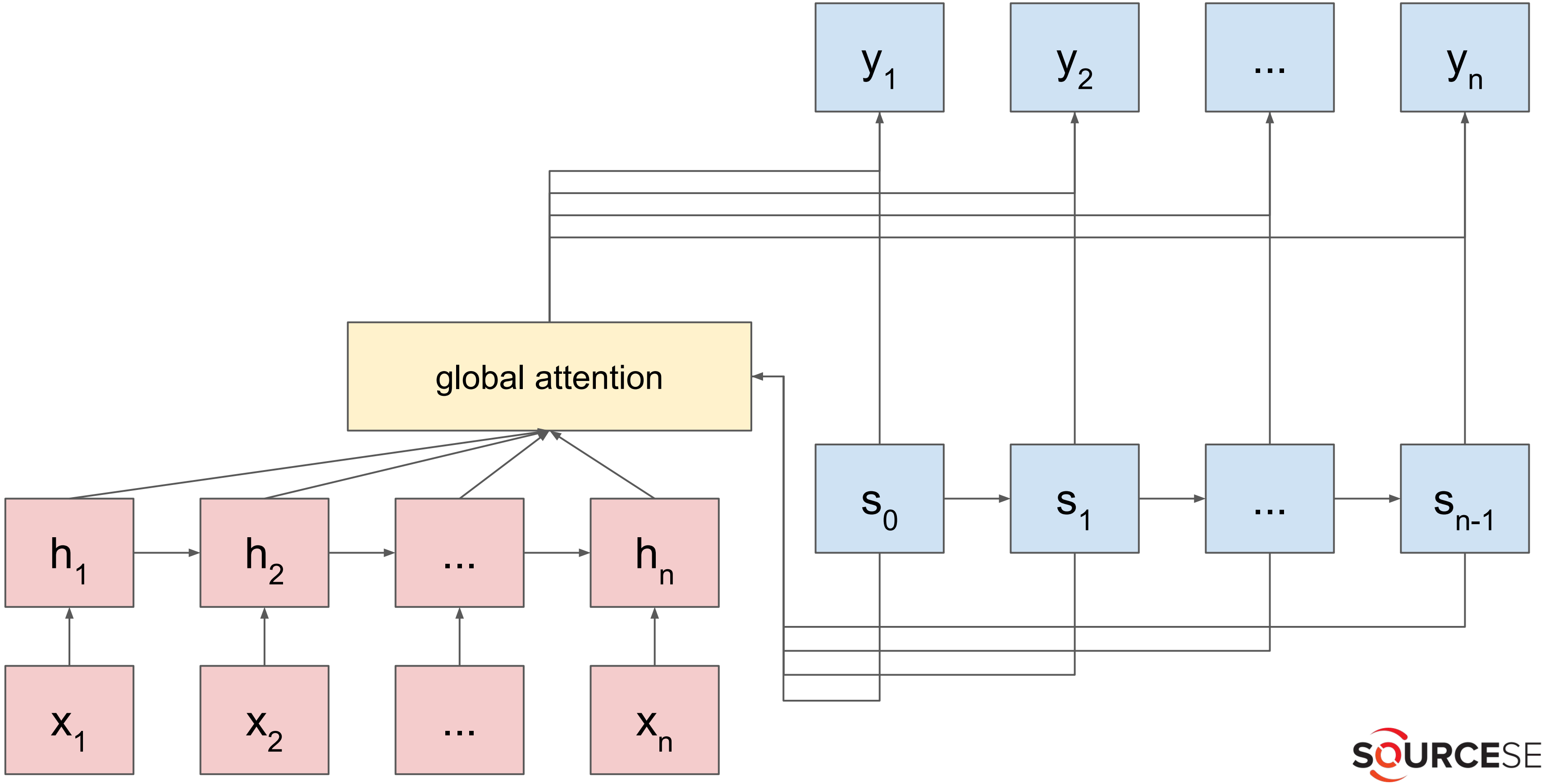
SOURCESENSE

Motivation for HAN-NMT

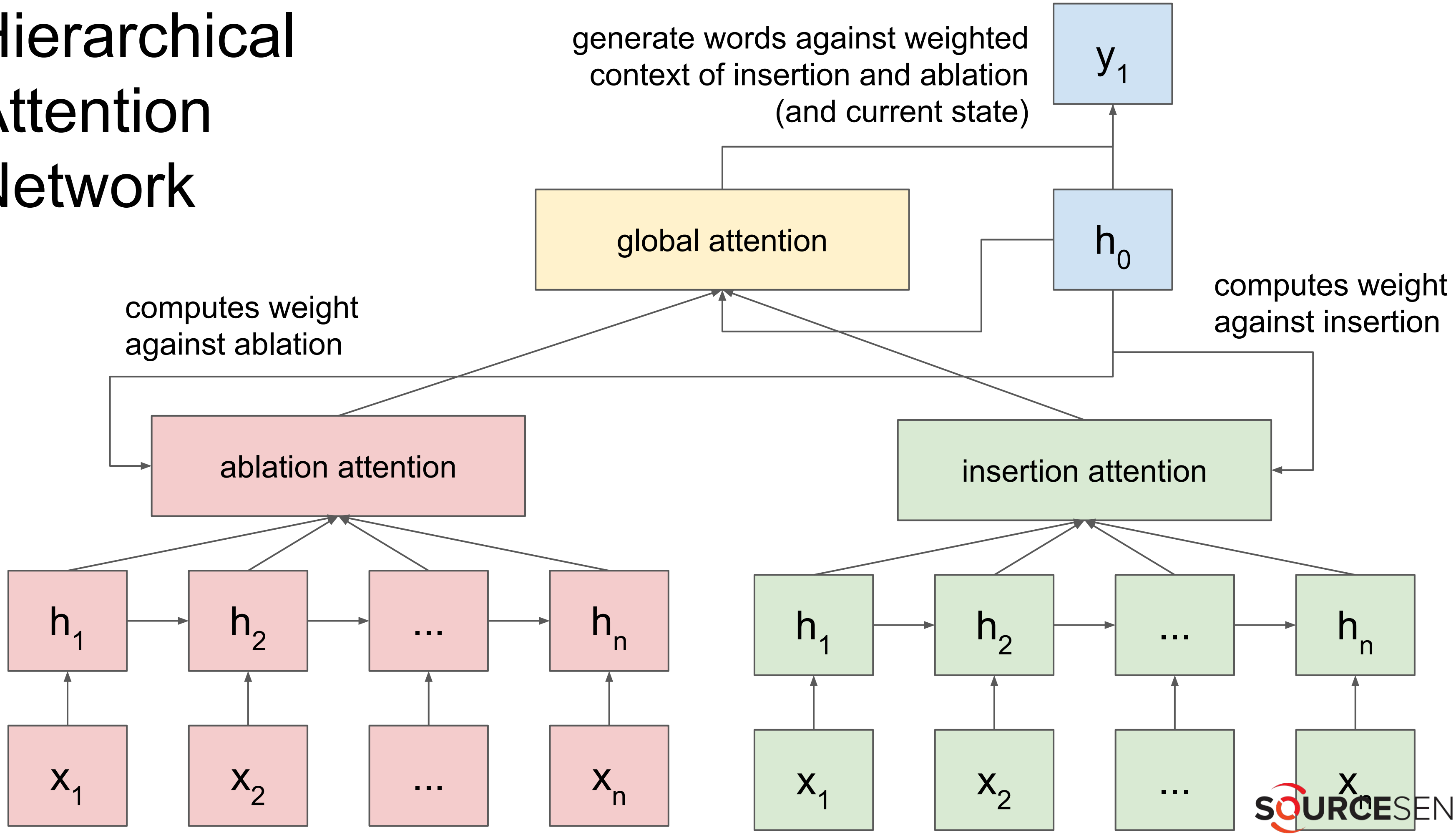


Input complexity is factored into separate contexts.
Speed in unimpacted (same number of matmul +3) but precision should improve.

Traditional attention



Hierarchical Attention Network



TO BE CODED...

Thanks for the attention

aijanai/vanilla-neural-commit-suggester