PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 104 OCTOBER 2015 27–38

Box: Natural Language Processing Research Using Amazon Web Services

Amittai Axelrod

www.BoxResear.ch

Abstract

We present a publicly-available state-of-the-art research and development platform for Machine Translation and Natural Language Processing that runs on the Amazon Elastic Compute Cloud. This provides a standardized research environment for all users, and enables perfect reproducibility and compatibility. Box also enables users to use their hardware budget to avoid the management and logistical overhead of maintaining a research lab, yet still participate in global research community with the same state-of-the-art tools.

1. Introduction

Amazon Web Services (AWS) is the umbrella term for all of the remote services that make up the Amazon cloud-computing platform. AWS includes the Elastic Compute Cloud (EC2) virtual computer cluster, as well as related services for storage, monitoring, analytics, and others.

We present our mechanism for using the Elastic Compute Cloud as the basis for a research and development platform for natural language processing researchers, students, and professionals. Using a cloud-based platform has numerous advantages. Compute nodes ("instances" in EC2 parlance) can start with the exact same software configuration, including installed tools, directories, and users. In particular, instances can be launched as direct clones of an existing virtual machine, or Amazon Machine Image (AMI). Our Box project creates such a disk image, containing open-source tools of use to the MT and NLP research community. As the disk images are frozen, identically-configured machines can be launched repeatedly, enabling stable and reproducible results. Future Box releases will be as separate AMIs, ensuring that

 Corresponding author: amittai.box@gmail.com
Cite as: Amittai Axelrod. Box: Natural Language Processing Research Using Amazon Web Services. The Prague Bulletin of Mathematical Linguistics No. 104, 2015, pp. 27–38. doi: 10.1515/pralin-2015-0011.

previous versions are always available, and previous experiments and code can always be re-run. Users have `sudo` access to instances they launch, so any Box machine can be updated or modified as desired.

While each instance has the same software configuration, each one can have a different hardware configuration, depending on the users' need. Box instances presently range from 1 vCPU with 3.75 Gb RAM (`m3.medium`) to 32 vCPU with 60Gb RAM (`c3.8xlarge`). The next major release of Box will expand to support a larger range, from 1 vCPU with 1Gb RAM through 32-40 vCPU with 160-244Gb RAM.

The downside is that while the compute nodes and the installed tools are open-source and free ("libre, as in software"), running the EC2 instances themselves is not free ("gratis, as in beer"). Nonetheless, cloud-based computing now enables researchers to spend their hardware budget on an as-needed basis, and scale their experimentation accordingly, rather than paying upfront to purchase, set up, and maintain a fixed physical cluster. As such, working on Box (or EC2 in general) can yield higher experimental throughput for a fixed budget.

Box itself comes with many SMT tools pre-installed, with the goal of enabling immediate productivity and lowering the barrier to entry. Furthermore, rather than setting up the complete pipeline for one MT toolkit, as do most research labs, Box comes with both the Moses (Koehn et al., 2007) and cdec (Dyer et al., 2010) pipelines, as well as compatible tools: fast_align (Dyer et al., 2013), mgiza (Gao and Vogel, 2008), kenlm (Heafield, 2011), and Jonathan Clark's ducttape. We will shortly include Joshua (Li et al., 2009), as well as other MT tool suites. By doing so, we hope to enable head-to-head comparisons of MT tools as well as mixing-and-matching. This provides a reason to avoid parallel development, and reducing duplicated effort.

Box also includes other non-core MT tools to stimulate exploration. This release includes multeval (Clark et al., 2011), rnnlm (Mikolov et al., 2010), and word2vec (Mikolov et al., 2013), with the intent to expand the list shortly. There are many open-source tools that each perform one NLP-related task well, such as speech recognition, optical character recognition, text-to-speech, dialog systems, and so on. However, there are few end-to-end open-source multilingual systems available for desirable tasks such as translating visual input from a smartphone camera, spoken announcements in subway stations, or dialog in real-time. We hope that by providing easy access to available components, we can accelerate progress.

2. Case Study

Box was intended from the beginning to be used by researchers without sufficient computational resources to work effectively. Lacking access to a cluster, the experimental results in our dissertation (Axelrod, 2014) were originally scoped for a single 4-core desktop machine. However, that allowed training only one or two MT systems per week; not conducive to training a thousand systems and graduating efficiently. We therefore architected a workflow to use Amazon's Elastic Compute Cloud

to mimic a traditional filesystem and cluster. For financial reasons, we also prioritized not leaving machines idle.[1] This workflow is independent of Box itself: each Box instance is also designed to be launched and used as a standalone development machine. What follows is only one possible way of using Box.

2.1. Instance Specifications

We primarily used three sizes of general-purpose Amazon EC2 instances. Specs for the current generation of instances are listed in Table 1.[2] It is worth mentioning that instance pricing appears to decrease slightly over time. A `medium` instance suffices for training a Moses system on TED-sized data (150k sentence pairs) in a few hours, for a negligible cost. A medium instance can also be used to run just the Moses decoder using translation models that have been filtered down to match a particular test set. A `large` instance is suitable for training up to around 1.5M sentence pairs, or most of Europarl, in under a day. For larger corpora we used the `xlarge` instance type, and found that the extra cores meant that 2M sentences trained faster on an `xlarge` than 1M on a `large`.

Name	vCPU	Memory (GB)	Hourly Cost
m3.medium	1	3.75	$0.067
m3.large	2	7.5	$0.133
m3.xlarge	4	15	$0.266

Table 1. Basic Amazon EC2 instance specifications.

2.2. Workflow

The Amazon Elastic Compute Cloud can be used effectively while only running the smallest (and cheapest) possible machine (a `*.micro` instance) full-time. The cluster nodes – termed EC2 instances – cost more, and thus are launched for particular experiments and terminated immediately afterward. The micro instance (the *home instance*) is a regular Linux machine, not a Box clone, and it acts as a gateway machine. The home instance is the white box in the upper right of Figure 1.

The home instance has a very large storage volume attached (volumes are only accessible if they are mounted to a running instance). This large volume, shown as a large circle in the lower right of Figure 1, acts as central filesystem storage for corpora

[1] If money is no object, then one could launch 20 EC2 instances, and run them year-round without worrying about idle nodes. However, it is likely cheaper to buy the hardware and hire a sysadmin.

[2] A Box instance costs 10% more to use per hour than a bare-bones instance, but all other Amazon fees for data transfer and storage are unchanged.

and experimental results. This avoids the need to upload or download large amounts of data to the Box instances. Amazon charges users to transfer data, and transferring data between EC2 instances is faster and cheaper than between EC2 and the non-Amazon world. As such, it is more efficient to transfer all data at once.

Corpora that are going to be passed to multiple Box instances should be stored in a data snapshot (shown as the grey circle in the lower left of Figure 1). This snapshot can then be cloned into new volumes for each experimental node. These cloned volumes are the white circles in the middle of Figure 1.

Amazon Machine Images (the AMI is the grey box in the upper left of Figure 1) are snapshots of computers. The AMI is used to launch new machines called *instances*, which are the white boxes in the middle of Figure 1. While the instances can have different hardware configurations, they all have the same installed software and setup.

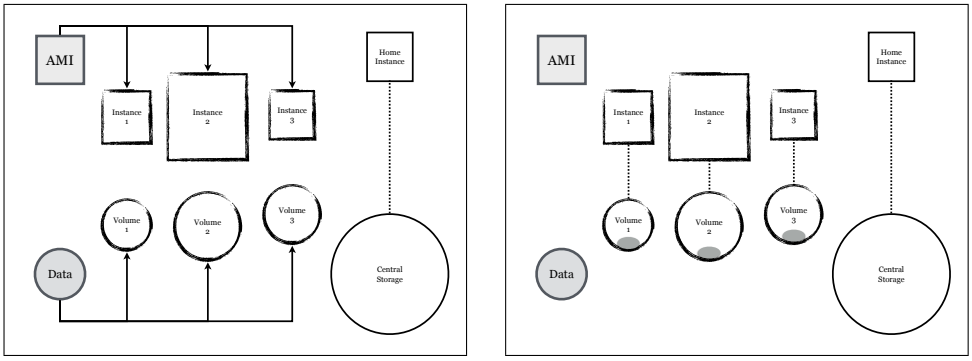

Figure 1. (Left) Launching instances from an AMI and cloning storage volumes from a data snapshot. (Right) After the jobs finish, experimental output is stored on the volume attached to each node.

The new volumes are attached to the new instances, and the experiments are run. Box instances are configured to be ordinary (yet powerful) computers, so experimentation can either be done by manually logging in to each instance and issuing commands in a normal session, or via writing the entire experiment as a single shell script and then executing the job on the Box instance. After the jobs finish, we store the experimental output on the volume attached to each node. The results are illustrated as grey content on the white data volumes in Figure 1. Having each instance work in a directory on an attached volume protects the results from accidental termination of the instance, which deletes all data stored on the instance itself. It is also a cost-saving

measure, as some cheap instances have little disk space included, and an additional volume is much cheaper than a larger instance.[3]

The experimental results are copied from the attached volumes to the central storage volume, and then the instances are killed and the volumes deleted, as shown in Figure 2.

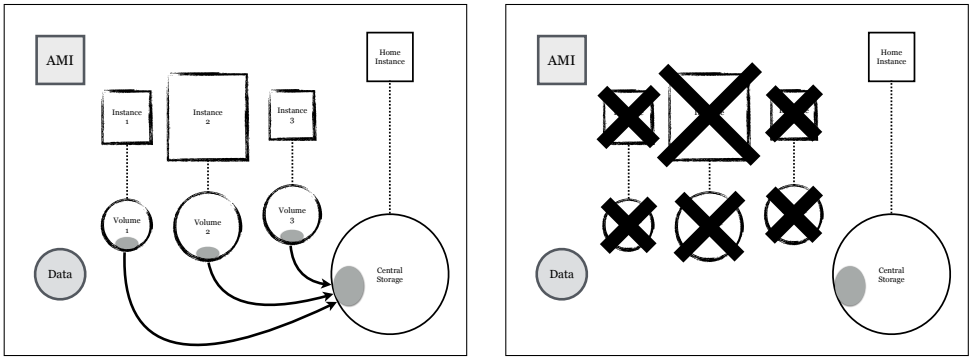

Figure 2. (Left) Experimental output is copied from each instance to central storage. (Right) All instances and attached volumes are deleted.

2.3. Reproducibility

We used the first (alpha) version of Box for all of the dissertation experiments, run over a two year span. While revising the final document, we were able to re-run experiments from 18 months prior and compare against newer work. This reproducibility is difficult to achieve even when a researcher has complete control of their development environment, as once tools are updated they are rarely rolled back even if they could be. With Box this is easy, as the development environment can be updated as needed, but older versions remain frozen yet in running order.

3. Signing up for the Elastic Compute Cloud

To work on the Amazon Web Services cloud, users need: an Amazon.com account, a telephone number (for account confirmation), and a method of payment and billing address. There is no charge to set up an account, and there is a year-long free usage tier

[3] One gigabyte-month of storage costs less than one compute-hour.

for new customers. There are how-to videos[4] for creating an AWS account, launching instances, and more.

3.1. Setup process

The setup process[5] must be done only once, and is as follows:

1. Make an AWS account.
   Visit `aws.amazon.com`, click "sign up",[6] and fill out the forms as instructed.
2. Create AWS security keys (optional).
   These are for using the command-line interface to the AWS account instead of the browser. They are not necessary if the user is prefers to use only the browser's GUI console to launch and manage their cluster usage. The names "Access Key" and "Secret Access Key" as used by Amazon may be confusing. They are simply a username/password pair that can be changed independently of the AWS account name and password. The keys can be generated from the Identity and Access Management (IAM) console, via User Actions → Manage Access Keys → Create Access Key[7].
3. Get SSH keys.
   This is a standard RSA keypair, necessary to log into EC2 instances. It is created via the EC2 console (`https://console.aws.amazon.com/ec2/`), under Navigation → Network & Security → Key Pairs.[8] The private key will automatically download – only once! – as a `*.pem` file. Failing to save the private key will render the ssh keypair useless and the process must be repeated. [9]
   The private key must have specific file permissions, and set as follows:
   ```
   chmod 400 $PRIVATE_SSH_KEYFILE
   ```

3.2. A Note on EC2 and Geography

Amazon's Elastic Compute Cloud is hosted in several locations around the globe, forming independent sets of resources. *Regions* are top-level geographical distinctions

[4] Video guides: `http://aws.amazon.com/getting-started/`

[5] Setup guide: `http://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-set-up.html`

[6] Or go directly to `https://portal.aws.amazon.com/gp/aws/developer/registration/index.html`

[7] IAM Console: `https://console.aws.amazon.com/iam/home?#home`
See `http://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-set-up.html#cli-signup`

[8] Creating SSH keys:
`http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html#having-ec2-create-your-key-pair`

[9] Note that the Amazon documentation refers to the SSH private key file as `my-key-pair.pem`. This implies the existence of an keypair called `my-key-pair`, but the `my-key-pair.pem` file contains *only* the private key and not the public one! This is not necessarily what might be expected. To avoid confusion, we refer to the keypair as `$SSH_KEYPAIR` and the private key file as `$PRIVATE_SSH_KEYFILE`, with the understanding that `$PRIVATE_SSH_KEYFILE = "$SSH_KEYPAIR.pem"`.

(*e.g.* "U.S. East", "South America"). By default, nothing is replicated across multiple regions, but any user can access any region regardless of where they are located, so the current version of Box is available only in the `us-east-1` region.The primary differences between EC2 regions are latency, and resource cost[10].

Availability Zones are subdivisions within a single EC2 Region, and named accordingly (*e.g.* `us-east-1a`, `us-east-1b`). By default Amazon will load-balance the availability zones, resulting in some instances launching in one availability zone, and some in another. It can be useful to have everything within a single availability zone to ensure complete interoperability, particularly if volumes will be each attached to and detached from multiple instances. The examples in this work assume the user is using the AWS `us-east-1` region and the `us-east-1a` availability zone, regardless of where in the world the user is physically located. The `us-east-1` region is required for the current release of Box, but can be changed to adapt the instructions to other AMIs. The choice of availability zone here is arbitary, and can be set at whim.

4. Signing up for Box

Accessing a community-made *paid AMI* such as Box requires first adding the associated *product* to the user's account.[11] This can be used to grant access to a number of related AMIs at once. At present there is only one Box AMI (`ami-1d678876`, v2015.05.26) in the product, but each future release will be a separate AMI. This ensures perfect reproducibility, as new Box AMIs provide expanded and updated research tools, but previous Box releases remain static and accessible. An experiment done once on a fresh Box instance can always be done on an instance of that Box.[12]

Amazon does the association of AWS account to Box product via a purchase for $0.00. As such, users are asked to confirm the payment method and shipping address from their AWS signup, though nothing is charged and nothing will be shipped. The link to sign up for Box is here: `https://portal.aws.amazon.com/gp/aws/user/subscription/index.html?offeringCode=49B2A70F`. Signing up for Box is free (as in beer). Only the actual use of Box incurs charges. If accessing other virtual machines on EC2, the cost-conscious researcher should note that other AWS EC2 products may have upfront fees or monthly charges in addition to (or instead of) usage costs.

[10] The `us-east-1` region has historically been the cheapest.

[11] Ironically, the Amazon server that handles these transactions is slow, and each page takes 30 seconds to load. Fortunately this process needs to be done only once.

[12] As long as the underlying Linux distribution of the Box (or any) AMI remains compatible with the hardware used by EC2. At present this is not an issue. All releases of the Amazon Linux AMI (since it exited Beta in 2011) are still runnable, even the 32-bit ones, indicating Amazon intends to preserve compatibility.

5. Basic AWS EC2 Operations

Few commands are necessary to use EC2 as a research resource, whether with Box or some other virtual machine. It is useful to know how to:

1. Launch a new instance from an existing Amazon Machine Image (AMI)
2. Create a new disk volume (optionally from a snapshot of an existing volume)
3. Attach a volume to an instance
4. Delete a volume
5. Terminate an instance

Of these, only launching and terminating instances are requirements. Being able to create, delete, and attach new volumes[13] is helpful for cloning disk volumes that already contain corpora, so as to avoid having to copy data over every time.

5.1. AWS EC2 Web Console

All of the instance management for EC2 can be done from the AWS EC2 console in a web browser[14] (Chrome, Firefox, Internet Explorer, and Safari are supported).

5.1.1. Launching a new Box instance via the console

The easiest method of launching a new instance is to use right-click on a running instance and select "Launch another instance like this one". The second-easiest method is to bookmark the URL of the launch wizard for a new Box instance:

`https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#LaunchInstanceWizard:ami=ami-1d678876`

The wizard presents the user with a series of steps:

Instance Type: For building SMT systems, it is important to pick an instance with enough memory for the size of the training corpus. Many configuration options can be found by scrolling down; greyed-out machine types are not available.[15]

Configure Instance The VPC (Virtual Private Cloud) *network type* is the default option and easiest for new users. Any *availability zone* will work, but creating all instances and volumes in the same zone is best (see Section 3.2). *Monitoring* is not useful.[16]

[13] Volumes detach automatically when the instance they are mounted on is terminated.

[14] EC2 Console: `https://console.aws.amazon.com/ec2/v2/home`

[15] Complete list of EC2 instance types: `https://aws.amazon.com/ec2/instance-types/`
Complete list of EC2 instance prices: `https://aws.amazon.com/ec2/pricing/`

[16] Monitoring tracks instance state, such as uptime of a customer-facing webservice, but increases costs.

Additional Storage Some instance types do not include much disk space. Disk is cheap, so it is easy to add any desired amount. However, all of this storage is deleted when the instance is terminated. A useful alternative is to not add additional storage to the instance directly, and instead create a new EC2 disk volume with plenty of space and attach it to the instance (see Sections 5.1.2 and 5.1.3). This prevents accidentally losing data or experimental results when the instance is terminated.

Tag Instance It can be difficult to differentiate instances (and attach a volume to the correct one, for example) when many are running. Unlike nodes in a regular cluster, each instance can be given a tag to indicate the experiment or process that is running on it. Prepending the date to the instance labels (*e.g.* "150524 building Box beta") can help keep the list of instances sorted.

Security Group Box instances are a good way to work on restricted or sensitive datasets, as corpora can be kept completely separate from other clients' or users' data. That being said, while crucial in fixed-location or long-duration applications, security settings are less important for a machine that will be terminated sooner rather than later. The default of "all ports, all sources" is good enough for casual users logging in with `ssh -i $PRIVATE_SSH_KEY.pem`. Diligent users can restrict access to port 22 (SSH) only, or permit only a particular port from a particular IP address.

After Launch The final step is to launch the instance and log in. See Section 5.3.

5.1.2. Creating a new storage volume via the console

The AWS console's sidebar has a section titled "Elastic Block Store". Clicking "Volume" → "Create Volume" will pop up a small wizard for making a new volume that will exist independently of any instances. For *Type*, "General Purpose SSD" will suffice. Select the size as needed, and the Availability Zone to match other instances and volumes already created. To clone a volume that already contains the desired corpora or models, enter the *Snapshot ID*. If the volume created is larger than the snapshot it is created from, then the volume will need to be resized with `sudo resize2fs` after it is attached to a running instance.

5.1.3. Attaching a volume to a running Box instance via the console

Independent storage volumes must be attached to – and then mounted on – running instances in order to be accessed. In the Volumes view of the EC2 Console, right-clicking any Volume ID produces a menu with the option to *Attach Volume*. Enter the target instance (the instance name tags from Section 5.1.1 are useful here). Note that the instance must be in the same Availability Zone. The last option is to select

a Linux device name for the volume. Single-use instances can generally be expected to have only one attached volume, so attaching all volumes to the same location (*e.g.* `/dev/xvdf`) allows all instances to run the same setup scripts. After attaching the volume, it must be mounted. This is done with `sudo mount` after logging in to the instance.

5.1.4. Deleting a volume

The same right-click menu on a Volume ID contains an option to *Delete Volume*. This should only be done after copying all experimental output off of the volume.

5.1.5. Terminating a Box instance via the console

The EC2 Console includes an Instances view, selectable via the sidebar. Right-clicking any Instance ID shows a menu which includes an option to *Terminate Instance*. This kills the instance immediately, deletes everything stored on the instance. Any attached volumes are unmounted and detached, and their contents are preserved.[17]

5.2. The Command Line Interface

Experienced researchers may appreciate being able to script these interactions via the Amazon Web Services Command Line Interface (AWS CLI) tool. The AWS CLI tool is itself released[18] under the Apache 2.0 licence, making it compatible with both derivative commercial works and the open source community. A detailed guide to using the AWS CLI for managing instances can be found on our website.

5.3. Logging in to a Box Instance

All EC2 instances take 2-5 minutes to boot after launch. After the instance finishes booting, its status will change to green in the console. The next step in using Box is to access the instance. Either its public IP address or DNS name, found by clicking on the instance in the EC2 Instances view, can be used to log in. The default username is `ec2-user`, thus:

```
ssh -i $PRIVATE_SSH_KEYFILE ec2-user@$IP_ADDRESS
```

Once the user is logged in to the instance, there are some optional steps to finish the instance setup. The first is to add some swap space, as by default there is none:

```
sudo dd if=/dev/zero of=/media/ephemeral0/swapfile bs=3M count=1024
sudo mkswap /media/ephemeral0/swapfile
sudo swapon /media/ephemeral0/swapfile
sudo swapon -s
```

[17] This is the advantage of using attached volumes when working on EC2 instances.

[18] AWS Git repo: `https://github.com/aws/aws-cli`

The second step is to mount any attached volumes:

```
mkdir -p ~/exp/
sudo mount /dev/xvdf ~/exp/
sudo resize2fs /dev/xvdf
```

Recall the suggestion that for ease of management, all volumes attached as in Section 5.1.3 be labeled `/dev/xvdf`. These commands can then be placed at the beginning of every experiment script and run automatically on all instances.

The Box instance is now ready for use.

6. Collaborating with Box

Multiple users can share a single Box instance. For a quick look around on an instance (*e.g.* to help debug a lab project on a student's Box), the `$PRIVATE_SSH_KEYFILE` file can be shared with the instructor. As the default username is the same for each EC2 instance (`ec2-user`), this is a very fast and simple way of sharing access to an instance. However, this presumes a trusted relationship already exists between the two parties. Sharing keys is otherwise anti-recommended for semi-permanent instances open to the outside world, or where multiple users' work may collide.

For longer or more formal collaborations, it is better to add new users. All Box instances are Linux machines with `sudo` privileges for the main user, so this can be done at will.[19] Each new user has their own private ssh key, and can log in separately just like any other server. The advantage, of course, is that granting access to a shared research instance does not involve granting access to an entire filesystem within the users' organizations. In this way it is possible for researchers to collaborate across institutional (or academic/industry) administrative boundaries, or to teach an online course to students who are not formally enrolled in the instructors' department.

7. Conclusion

The current release of Box provides access to a state-of-the-art research and development environment to both experienced and new researchers alike. Box runs on the Amazon Elastic Compute Cloud, so it can be used to provide computational resources to those who have none, or to supplement an existing cluster. Standard open-source toolkits for machine translation and natural language processing are pre-installed on Box, putting any user in the position to start work immediately. This enables students, researchers, and developers to contribute to the field without being limited by their computational resources at hand. By this mechanism we hope to substantially lower the barrier to entry for the field of NLP.

[19] Adding users: `http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/managing-users.html`

Acknowledgements

The author appreciates the early encouragement of Achim Ruopp, who made the first Moses installation for EC2.

Bibliography

Axelrod, Amittai. *Data Selection for Statistical Machine Translation*. PhD thesis, University of Washington, 2014.

Clark, Jonathan H, Chris Dyer, Alon Lavie, and Noah Smith. Better Hypothesis Testing for Statistical Machine Translation : Controlling for Optimizer Instability. *ACL (Association for Computational Linguistics)*, 2011.

Dyer, Chris, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blumson, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. cdec: A Decoder, Alignment, and Learning Framework for Finite-State and Context-Free Translation Models. *ACL (Association for Computational Linguistics) Interactive Poster and Demonstration Sessions*, 2010.

Dyer, Chris, Victor Chahuneau, and Noah A Smith. A Simple, Fast, and Effective Reparameterization of IBM Model 2. *NAACL (North American Association for Computational Linguistics)*, 2013.

Gao, Qin and Stephan Vogel. Parallel Implementations of Word Alignment Tool. *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, 2008.

Heafield, Kenneth. KenLM : Faster and Smaller Language Model Queries. *WMT (Workshop on Statistical Machine Translation)*, 2011.

Koehn, Philipp, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Christine Moran, Chris Dyer, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. *ACL (Association for Computational Linguistics) Interactive Poster and Demonstration Sessions*, 2007.

Li, Zhifei, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Sanjeev Khudanpur, Lane Schwartz, Wren N G Thornton, Jonathan Weese, and Omar F Zaidan. Joshua: An Open Source Toolkit for Parsing-based Machine Translation. *WMT (Workshop on Statistical Machine Translation)*, 2009.

Mikolov, T, M Karafiat, L Burget, J Cernocky, and S Khudanpur. Recurrent Neural Network based Language Model. *INTERSPEECH*, 2010.

Mikolov, Tomas, Quoc V Le, and Ilya Sutskever. Exploiting Similarities among Languages for Machine Translation. *arXiv preprint arXiv:1309.4168v1*, 2013.

Address for correspondence:
Amittai Axelrod
amittai.box@gmail.com
4423 Lehigh Rd #666,
College Park, MD 20740, USA