基于 Moses 的机器翻译引擎构建与部署应用研究*

李峰[1,2]，黄金柱[3]，张克亮[3]，曾荣仁[1]

（1. 中央军委后勤保障部后勤科学研究所，北京 100166；

2. 北京航空航天大学外国语学院，北京 100191；

3. 解放军外国语学院语言工程系 河南洛阳 471003）

摘要：本文首先对机器翻译引擎的构建思路进行了全面的规划，简要介绍了 Moses 3.0 系统及其特性，理清了引擎的构建思路、形成了引擎构建的总体规划。随后把机器翻译引擎的构建与部署划分为四个阶段进行了详细的描述，包括系统环境与语料的准备、三个核心模型的构建、多个参数的调优与系统的整体优化和引擎部署时硬件资源的预估和部署方式，然后基于大规模的中英语料构建了一个真实可用的机器翻译引擎，最后从机器翻译引擎的可用性、翻译结果质量、翻译效率三个方面对本文构建的引擎进行了详细的评估，其中在翻译质量评估时与百度、有道、微软等在线机器翻译系统进行了实验对比测试，并给出了可能导致实验结果的原因。

关键词：统计；机器翻译；Moses 3.0；部署与应用

中图分类号：TP391　　**文献标识码：**A

The Study of Constructing and Deployment of Moses-based MT Engine

Li Feng[1,2], Huang Jinzhu[3], Zhang Keliang[3], Zeng Rongren[1]

(1. Logistics Science Research Institute of Logistics support department of the Central Military Commission, Beijing 100166, China;

2. The school of foreign languages of Beihang University, Beijing 100191, China;

3. Department of Language Engineering of PLA University of Foreign Languages, Luoyang 471003, China)

Abstract: Firstly, the paper makes an overall planning over the MT engine construction which covers brief introduction about Moses system and its characteristics, the process and general planning of constructing the MT engine. Based on the above, the paper divides the design and implementation depiction of the MT engine into four parts which covers the preparation of system environment and corpus, the construction of three core models, parameter and MT system optimization, the estimation and deployment of hardware resources in the process of MT deployment. Then, the study, based on large-scale corpus, implemented a usable Machine Translation engine, and further makes a detailed evaluation from the aspects of usability, translating quality and efficiency upon the engine. Concerning translation quality, the paper carries out comparative testing with some of famous online MT system such as Baidu, Youdao and Microsoft etc., and the possible reasons leading the testing results are also given.

Key words: Statistics; Machine Translation; Moses 3.0; Deployment and Application

1 引言

使用机器进行翻译的思想起源于使用第一批计算机用来解读战争中的译码[1]，经历几十年的曲折发展，随着全球一体化以及国家间的各种往来日益密切，对机器翻译的需求愈加强烈，机器翻译技术也取得了长足的进展。其中，基于规则、基于实例与基于统计的机器翻

* **基金项目：**国家自然科学基金（61370126）；国家社会科学基金（15GJ003-154）；软件开发环境国家重点实验室探索性自主研究课题基金（SKLSDE-2015ZX-16）

作者简介：李峰（1982—），男，博士，主要研究方向为自然语言处理、机器学习；黄金柱(1980—)，男，博士生，主要研究方向为自然语言处理、本体资源库建设；张克亮（1964—），男，教授，博士生导师，主要研究方向为自然语言处理、本体资源库建设等；曾荣仁（1973—），男，高级工程师，主要研究方向为大数据与智能计算。

译一直是本领域研究的主流[2,3]，近些年来，随着海量数据的出现以及计算资源成本的下降，基于统计的机器翻译研究开始占据绝对的优势，越来越多的研究者把目光转向了基于数据驱动的机器翻译研究，开源统计机器翻译引擎，如 Moses[4]、Joshua[5]和我国丝路机器翻译系统等的出现，更是助推了这一发展趋势。当前，谷歌、百度、微软、有道等企业先后上线了实时的在线机器翻译引擎，这些翻译引擎无一例外都是采用了基于统计的机器翻译方法。本文即将构建和部署的机器翻译引擎也是基于统计的，故而后文所有描述均围绕统计机器翻译引擎进行。

在基于统计的机器翻译流程中，通常需要使用大量的数据来训练相应的模型，其中最为关键的即句子层面对齐的双语平行语料。为了使有限的语料尽可能覆盖更多的现实场景中的句子，需要把对齐后的平行语料切割成更小的语言单元，如短语或词。通过这些更小语言单元的翻译拼接，组成最终翻译完成的句子。在对语料进行切割时，为了使翻译更加对应，尽可能地保持语义上的一致，需要基于这些平行语料进行语言单元的对齐，通常称之为词对齐[6]。词对齐确定了双语数据中源语言句子和目标语言句子之间的基本翻译关系，是统计机器翻译得以进行的基础。此外，在统计机器翻译中还有两个模型也至关重要：一个是语言模型[7,8]，该模型的作用是评价译文的流利度，目的是从候选译文中选择更加流利的输出；另一个是调序模型[9,10]，不同语言之间语言片段的顺序往往不一致，这个模型主要是将目标语言片段按照语言习惯重新排列为合适的语序。

一个完整的统计机器翻译引擎除涉及上述三大模型外，还涉及语言与技术的方方面面，是一个跨多学科的综合性研究。几十年来国内外众多学者从机器翻译引擎的各个环节、各个细节着手展开了十分深入而详尽的研究，如模型构建、词对齐、参数训练[11]、译文质量评价[12,13]以及机器翻译引擎构建等，取得了显著的成绩，推动了机器翻译研究的持续发展。本文的研究着眼如何利用已经取得的最新研究成果，构建和部署一个完整的机器翻译引擎，是一项把科研成果向实践应用转化的尝试，在机器翻译引擎构建的过程中，注重把握各个环节的运行原理，力求使用最新的技术达到较好的效果，限于篇幅却不执拗于繁杂的技术细节。

下文首先简要介绍了 Moses 系统，然后给出了机器翻译引擎构建的思路与规划；接着描述了构建过程的各个关键环节，并使用大规模的平行语料在 Ubuntu 系统上构建了一套时政类中英机器翻译引擎；最后选取了百度在线翻译、有道在线翻译、微软在线翻译作为参考对比，对构建完毕的机器翻译引擎进行了评估和测试。

2 机器翻译引擎构建思路与规划

2.1 Moses 系统简介

Moses 系统是一套由英国爱丁堡大学统计机器翻译团队研发的开源统计机器翻译系统，当前最新版本为 3.0 版，使用 C++作为系统的主要开发语言[1]。该系统支持基于短语的机器翻译、基于层次短语的机器翻译和基于语法的机器翻译。同时，它提供了因子化翻译模型(factored translation model) [14]使得系统能够融入更多的语学信息，以提高翻译的质量。由于该系统采用了开源的模式，经过十余年的发展，通过不断吸收最新的机器翻译研究成果和不断的改进和升级，目前该系统在翻译性能和翻译质量上处于较为领先的水平。

Moses 系统兼容多种语言模型训练工具，如 irstlm[15]，Kenlm[7]；支持多种词对齐工具，如 GIZA++[16]，MGIZA[17]等；提供了许多有用的工具包，如平行语料噪音清洗、大小写转换、在线机器翻译引擎构建组件等模块或工具包，并提供了完整的操作说明文档，为研究者开发了专用的机器翻译实验系统，同时提供了少量的机器翻译实验语料和相应的机器翻译质量调优和评测包。通过采用最新的研究成果、提供丰富的配套工具、详尽的说明和帮助文档，目前该系统已成为机器翻译领域最有影响力的开源机器翻译系统。本文后续的研究也将基于该系统来构建和部署中英机器翻译引擎。

2.2 引擎的构建思路

本文在机器翻译引擎构建的过程中，采用先整体把握机器翻译引擎构建的各个关键环节，将全部过程规划为几个关键部分；再结合流程的各个组成部分和功能进行深入的研究；最后通过各个环节的对接与组合形成整个翻译引擎，并部署测试。其中，在规划时注重各个环节间的相对独立，同时保持环节间的有机衔接，例如语料的准备和翻译模型的训练这两个环节在操作上互相独立，但前者又是后者进行的基础；在具体到每个模块时，注重深入探究所选子系统的效率，尽可能保证系统的每个部件都能够较好、较快又较稳定地工作，例如，深入研究各个语言模型训练工具的特性，选择支持大数据量的 irstlm 语言模型工具和支持多线程的 Kenlm 语言模型工具进行语言模型训练等；据此达到整个机器翻译引擎的系统鲁棒性、运行高效率和译文高质量。

[1] Moses 3.0: http://www.statmt.org/moses/

2.3 引擎的构建规划

结合上文机器翻译引擎构建的思路，本文对机器翻译引擎的构建流程进行了整体规划。从各个环节负责的主要功能和环节间的相互关系着手，我们把机器翻译引擎的构建划分为七个环节，分别为：(1)引擎构建与部署平台的选用；(2)系统组件的安装与编译环境的准备；(3)语料的准备与预处理；(4)语言模型的训练；(5)Moses 内核的编译与翻译模型的训练；(6)参数调优与系统的优化；(7)硬件的预估和引擎的部署。这七个环节决定了系统的运行环境、使用的子系统以及系统的主要功能等，总体规划如图 1 所示：

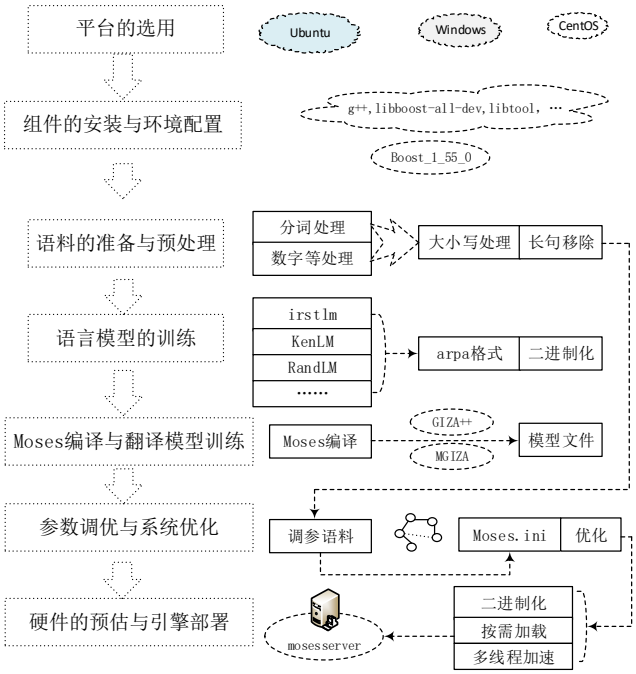

图 1 机器翻译引擎构建总体规划图

在基于 Moses 构建统计机器翻译引擎时，涉及到的诸多模块与子系统对应的应用平台主要为 Linux 系统，考虑到各个模块的系统兼容性和部署的便捷性，本文选择 Ubuntu 14.04 LTS 作为机器翻译引擎的编译与部署平台，并预先安装了各个模块依赖的系统组件，配置了相应的环境。

基于统计的机器翻译引擎最终翻译的质量取决于两个因素，一个是技术模型的先进性，一个是用于模型训练的语料质量和规模。在实际应用场景下通常需要 200 万以上的不重复句对作为翻译模型的训练语料，才能够取得相对较好的翻译质量。

语言模型的训练主要是通过分析对比现有的语言模型训练工具，如 IRSTLM、Kenlm 等，根据引擎构建的需要选择一种或多种语言模型工具来训练译文的语言模型。

本文选择 Moses 3.0 来构建机器翻译引擎的内核，使用 Giza++或 MGIZA 词对齐工具进行翻译模型和调序模型的训练，并对训练得到的模型文件进行二进制压缩，以提高系统运行的效率。

机器翻译引擎参数的调优主要是通过选取一定数量的语料，对翻译模型和调序模型中使用的各个参数进行不断调试，最后保留一组能够取得相对较优翻译结果的参数。

最后一个环节是机器翻译引擎的部署，这个过程涉及各个模型的加速，例如二进制化翻译模型、调序模型等，并根据实际应用情况和要求对所需的硬件资源进行预估，最后结合 Ubuntu 系统开放翻译服务端口提供翻译服务。

3 机器翻译引擎构建的关键流程

根据上文规划的七个主要环节，本文将机器翻译引擎的构建流程划分为四个关键步骤：(1)系统环境与语料的准备；(2)核心模型的构建；(3)参数的调优与系统优化；(4)硬件资源的预估与翻译引擎的部署。其中，系统环境与语料的准备是基础，核心模型的构建是主体，参数的调优与优化是提高翻译效率与质量的有效途径，而硬件资源的预估与翻译引擎的部署是研究成果向实践应用转化的桥梁。在关键流程的描述中，本文以构建一个面向生产环境的中英机器翻译引擎为例进行说明。

3.1 系统环境与语料的准备

系统环境与语料的准备是整个机器翻译引擎构建的基础，其中语料的规模初步决定了系统环境的硬件配置，而系统环境的硬件配置和软件配置共同决定了机器翻译引擎各个模块是否可以顺利编译并高效运行。前期的准备工作主要包括：硬件资源配置、系统环境配置和平行语料准备三个部分。

3.1.1 硬件资源配置

面向生产应用环境的机器翻译引擎的构建与运行通常需要较高的硬件资源支撑。鉴于在初期的构建过程中，仅是要求能够满足训练和测试，因而采用了较低的硬件配置：

CPU：i7 4790 8 核处理器，内存：32G 1600MHz，硬盘：2T。

3.1.2 系统环境配置

本文采用 Ubuntu 14.04 LTS 作为机器翻译引擎的承载系统。在该环境下，Moses 系统要求在系统运行之前必须进行相应组件的安装以及编译环境的配置。必须安装的组件包括：

g++、*git*、*subversion*、*automake*、*libtool*、*zlib1g-dev*、*libboost-all-dev*、*libbz2-dev*、*liblzma-dev*、*python-dev*、*libgoogle-perftools-dev*、*cmake*

在组件安装完毕后，需要编译安装 Boost，此模块是 Moses 翻译引擎得以成功编译和运行的基础，本文采用 boost 1.55.0 版本进行编译，操作命令为：

./bootstrap.sh

./b2 -j4 --prefix=$PWD --libdir=$PWD/lib64 --layout=system link=static install || echo FAILURE

其中 j4 表示的是 CPU 的核心数。

以上系统环境配置为 Moses 机器翻译引擎得以运行的必需条件，任何一个组件的编译或安装失败，均可能会导致系统无法正常运行。

3.1.3 平行语料准备

本文使用约 650 万不重复的中英翻译平行句对作为机器翻译引擎的构建语料，语料体裁覆盖日常生活的各个方面，如汽车、铁路、基建、IT、医疗、能源等领域。在各类模型构建之前，需要对语料进行预处理，主要包括四个部分：

一是对语料进行分词处理，本文采用 ansj 分词组件进行中文分词，采用空格对英文进行切分；二是进行大小写处理，这里使用 Moses 系统自带的大小写模型训练与校正工具 train-truecaser.perl 和 truecase.perl 来进行英文的大小写处理；三是日期、数字与网址的泛化处理，把句子中出现的这几类字符替换为统一的占位符，减少模型训练过程中的噪音；四是移除语料中过长的句子，避免过长句子带来的训练复杂性及模型干扰，这里依然使用 Moses 系统自带的 clean-corpus-n.perl 工具来处理，通常 n 取值为 40 至 100 间的值，本文在实际训练中取 80，该值表示源语言与目标语言中的句子长度均不得长于 80 个字符。

3.2 核心模型的构建

一个完整的统计机器翻译引擎通常包括语言模型、翻译模型与调序模型三个部分。

3.2.1 语言模型的构建

语言模型主要用来评价译文的流畅程度，最常用的语言模型是 n 元语言模型，其假设一个词的出现概率只与前 n-1 个词有关[18]。当前语言模型研究已相对成熟，但随着深度学习的广泛应用，也有学者提出了基于神经网络的语言模型，并认为基于该模型能够改进机器翻译的结果[19]。Moses 系统支持多个语言模型训练工具训练的结果，如 irstlm、Kenlm 等。其中 irstlm 是一个免费且开源的语言模型训练工具，其能够在大规模语料场景下通过采用划分词典分块训练合并的方式加速语言模型的生成，且能够保持较低的系统资源消耗，但该工具并不是完全线程安全的；Kenlm 模型为 Moses 自带的语言模型训练工具，也是免费且开源的，同样能够在较低的资源配置下保持较好的训练效率，较之于 irstlm 该语言模型支持 Moses 的多线程处理，且是完全线程安全的。本文联合使用 irstlm 5.80.08 和 Moses 3.0 自带的 Kenlm 语言模型构建工具来建立相应的语言模型。

语言模型的建立主要包括工具的编译、语料预处理、语言模型的训练、编译为 arpa 格式以及二进制化语言模型五个部分，其中前四个部分必须执行，而二进制化则为了提高语言模型的应用效率。限于篇幅，这里不再介绍相应工具的编译和语料的预处理方法，仅给出语言模型训练、arpa 格式语言模型的生成和语言模型二进制化的方法。

语言模型的训练采用 irstlm 工具可执行目录下的 build-lm.sh 工具进行，在训练结束后，使用同一目录下的 compile-lm 工具，对语言模型进行编译处理；最后采用 Kenlm 语言模型工具对生成的 arpa 格式的语言模型进行二进制化处理，在二进制化的过程中，可以采用 trie 化的数据结构来存储，也可以采用概率化的数据结构来存储语言模型，区别在于使用 trie 化的数据结构来存储时，可实现语言模型的按需加载。整个过程，前两步主要应用 irstlm 工具模型训练高效的优势，后面采用 Kenlm 对语言模型进行二进制化处理，目的在于提升语言模型加载与运行的效率，同时提升多线程处理时线程的安全性。

3.2.2 翻译模型与调序模型的训练

翻译模型根据翻译基本单元和建模方式的不同，可分为基于词汇、基于短语与基于句法的模型[18]。其中，基于词汇的模型为早期的机器翻译模型，因取得的翻译质量较差，现在已基本不用；基于短语的翻译模型以短语作为翻译的单位，在翻译模型中直接记录了常见的翻译对应模式和局部调序信息，因其模型简单具有较好的鲁棒性，且翻译过程中解码效率较高，得到的翻译质量也相对较好，成为机器翻译中的主流翻译模型；基于句法的翻译模型包括基于形式语法的翻译模型和基于句法树的翻译模型，前者利用形式语法等来实现翻译模型，比如基于层次短语的翻译模型，后者基于句子语法结构的自动分析来建立翻译模型，比如串到树模型、树到串模型和树到树模型。基于句法的翻译模型在较小规模的应用时，效果较好，缺点是过度依赖于句法分析的结果，同时模型解码效率很低，难以应用于高负荷的生产应用

场合。调序模型则主要用于处理不同语言间的语序差异，常用的调序模型包括词汇化的调序模型、基于多种稀疏特征的调序模型和基于深度神经网络的翻译模型[18]。

翻译模型和调序模型在基于短语的机器翻译模型，往往同步训练，而在基于句法的翻译模型中，因其本身已经包含了句法结构等调序信息，可以不用再训练专门的调序模型。

本文在构建中英机器翻译引擎时采用基于短语的翻译模型，并使用 Moses 系统的训练工具同步训练得到调序模型，主要步骤可分解为内核模块的编译、翻译模型与调序模型的训练两个关键部分。

(1)内核模块的编译

Moses 内核模块的编译主要包括三个步骤：

首先，需要为 Moses 指定编译变量。包括编译环境、语言模型配置、模型压缩参数配置、服务组件配置等，例如下面的编译命令：

./bjam --with-boost=/home/buaa/boost_1_55_0

--with-cmph=/home/buaa/cmph

--with-xmlrpc-c=/home/buaa/xmlrpc

--with-irstlm=/home/buaa/irstlm -j4

其中，--with-boost 表示编译环境参数；--with-cmph 表示模型压缩参数；-with-xmlrpc-c 表示服务应用参数；--with-irstlm 则表示语言模型参数；-j4 表示使用 4 个核心的 CPU 来支持 Moses 的运行。

其次，对 Moses 编译的全局检查。因 Moses 内核涉及的组件与模块较多，为避免出错，对 Moses 编译结果进行检查是机器翻译引擎成功构建的保证。具体方法，可使用 Moses 官方的测试样例包，执行如下命令：

~/mosesdecoder/bin/moses -f phrase-model/moses.ini < phrase-model/in > out

若在 out 文件中得到了 in 文件对应翻译结果，则表示编译全部成功，否则需要检查相应的错误，重新编译。

最后，是对 Moses 所用的语言模型进行测试。该步骤是保证语言模型能够为 Moses 正确使用的关键，相关命令形如：

echo "this is a English sentence ." | ~/mosesdecoder/bin/query ./news-commentary-v8.zh-en.blm.en

测试成功后会返回如图 2 所示的结果。

从图 2 中可以看出，该句话加上空格总共有 7 个单词，其中非词表词 0 个。

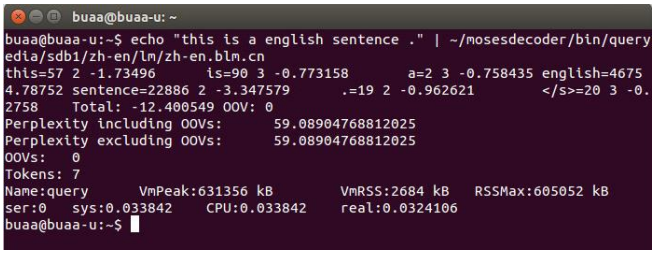

图 2 语言模型测试示意图

(2)翻译模型与调序模型的训练

翻译模型与调序模型的训练，在基于短语的机器翻译引擎中，可以同步生成。在模型训练时，本文采用 MGIZA 作为词对齐工具，该工具较之于 GIZA++的优势在于其能够使用多个线程进行同步操作，能够大大减少翻译模型和调序模型训练的时间。这里使用 Moses 系统自带的语言模型训练工具 train-model.perl 来训练翻译模型与调序模型，参考命令为：

*nohup nice ~/mosesdecoder/scripts/training/train-model.perl *

*-root-dir train-mgiza *

*-mgiza -mgiza-cpus 8 -cores 10 *

*-parallel -sort-buffer-size 10G -sort-batch-size 253 *

*-sort-compress gzip -sort-parallel 10 *

*-snt2cooc snt2cooc.pl *

*-corpus ~/corpus/zh-en.clean *

*-f zh -e en -alignment grow-diag-final-and -reordering msd-bidirectional-fe *

*-lm 0:3:$HOME/lm/zh-en.blm.en:8 *

-external-bin-dir ~/mosesdecoder/tools-mgiza >& training-mgiza.out &

上述命令表示在训练时，使用 MGIZA 进行词对齐，对齐过程中分配 8 个核心的 CPU 启用 10 个线程；在排序过程中，一次性分配 10G 内存用于缓存，并开 10 个线程用于加速计算。-alignment 和-reordering 则分别指定了翻译模型与调序模型训练时的参数。

3.3 参数的调优与系统优化

在机器翻译引擎的核心模型构建完毕后，理论上可以提供翻译服务了，然而此时的机器翻译引擎还没有优化，无论是翻译质量还是翻译速度都处于相对较差的状态，这就需要我们针对建立的系统进行参数的调优，并从系统层面进行优化，从而提高翻译的质量和速度。

3.3.1 参数的调优

在机器翻译引擎核心模型构建完毕后，系统会自动生成一下翻译服务配置文件 moses.ini，在这个文件中，保存着最为原始的参数。通常需要根据翻译服务对象，

对这些参数进行调试和优化，以从整体上提高翻译结果的质量。对参数的调优通常包括几个步骤：

(1)调参语料的准备。一般使用与翻译模型训练语料领域一致的，但并不包含在翻译模型训练语料中的其他语料来调参。在调参时，同样涉及前文所述的语料的预处理工作，如分词、大小写、泛化处理等，限于篇幅这里不再赘述。

(2)调参命令的执行。通常使用 Moses 系统自带的工具 mert-moses.pl 执行调参，参考命令形如：

*nohup nice ~/mosesdecoder/scripts/training/mert-moses.pl *

*~/corpus/test2015.true.zh ~/corpus/ test2015.true.en *

*~/mosesdecoder/bin/moses train/model/moses.ini --mertdir ~/mosesdecoder/bin/ *

*--decoder-flags="-threads 8" *

&> mert.out &

其中，命令中第二行为调参语料，第三行为等待调参的配置文件 moses.ini 及调参优化后的配置文件保存目录，最后一行指出了调参过程中使用的并发线程数。

值得指出的是，这个过程会消耗大量的系统资源，同时也非常耗时。采用前文所述的硬件资源配置及语料规模，调参耗时约在 2 天左右。

3.3.2 系统的优化

基于统计的机器翻译引擎在大规模语料环境下，意味着海量的数据计算，降低资源消耗提升翻译效率也是机器翻译引擎构建与部署实践过程中必须研究的问题。在系统资源硬件一定的条件下，通常通过两种方式来进行系统优化：一种是优化机器翻译核心模型的运行速度，另一种是通过在翻译过程中附加额外参数，降低计算的复杂度。

(1)优化核心模型提升运行速度。通常可对各个模型进行二进制化处理，来提升模型加载及运行的速度，例如针对翻译模型，可以使用 processPhraseTableMin 工具，按照如下命令样式实现二进制化处理：

*~/mosesdecoder/bin/processPhraseTableMin *

*-in train/model/phrase-table.gz -nscores 4 *

-out binarised-model/phrase-table

其中 phrase-table.gz 为原始翻译模型文件，phrase-table 为二进制化处理后的翻译模型文件。在对模型文件进行二进制化处理后，能够数倍提升模型的加载及运行速度。可以使用类似的方法对语言模型和调序模型进行二进制化。

在二进制化之后，需要在 moses.ini 配置文件中更改模型的类型及存储路径，使机器翻译能够在新的模式下提供翻译服务。

(2)使用参数加速翻译过程。除了通过二进制化核心模型以提升系统的翻译效率外，还可以使用额外的参数来限制计算的复杂度，从而加速翻译。例如，可以使用-ttable-limit 参数来限制翻译模型中短语表的搜索程度，使用-s 限制翻译时查找堆栈的大小，以及使用-b 限制柱状搜索，使用-dl 来限制调序过程中的扭曲程度。例如如下命令：

echo "我 是 一名 北航 的 学生." | ~/mosesdecoder/bin/moses -f ~/working/binarised-model/moses.ini -dl 5

其中，-dl 参数表示调序过程中可以扭曲的最大单词数，该值越小允许调序的范围就越窄，同时翻译速度也就越快，设值为 0 表示不调序，值为-1 时为不限制范围的调序。

3.4 硬件资源的预估与翻译引擎的部署

在真实的机器翻译应用场景下，必须根据实际需要对所需要的硬件资源进行预估，并在此基础上完成翻译引擎的部署，以保证能够提供实时的翻译服务。

3.4.1 最低硬件资源估算

在构建机器翻译引擎时，通常需要使用高性能的单独的服务器，即该服务器仅供翻译引擎使用，而不再提供其他服务。为了保证基于 Moses 构建的机器翻译引擎在构建完成后能够成功运行，需要对系统内存进行最低估算，一般方法为：

引擎能够运行的最低内存大小为前文所述语言模型、翻译模型及调序模型文件大小之和，再加上 500M 至 1G 容量的 Moses 系统运行内存。

例如，三个模型的大小之和为 5G，则系统最低内存为 6G 左右，否则系统无法加载和运行。

在真实应用场景，为了保证机器翻译引擎能够顺利流畅运行，通常内存最低量需要翻上好几倍，同时对 CPU 也有着较高的要求；在高并发场合下，可以通过大幅提升硬件资源，或使用多个翻译服务器轮循调度的方式实现高负载的翻译服务。

3.4.2 翻译引擎的部署

在机器翻译引擎构建完毕后，为进一步提高用户使用的友好性，需要对其进行部署，并提供翻译服务接口，而不是让用户在 Ubuntu 系统的命令行下逐句进行翻译。

Moses 系统提供了机器翻译引擎的部署程序，他通过在指定的端口提供 xml-rpc 服务的方式实现系统的打包与部署，涉及到的工具主要是 mosesserver，部署的方

法是在命令行下输入如下命令：

~/mosesdecoder/bin/mosesserver -f ~/working/binarised-model/moses.ini --server 192.168.128.140 --server-port 8080 --server-log ~/server.log

其中，moses.ini 为翻译服务器保存参数的配置文件，192.168.128.140 为当前机器翻译引擎所在主机的 IP 地址，8080 为监听翻译服务请求的端口，server.log 为翻译请求及应答日志文件。

至此，本节通过四个部分完整描述了基于 Moses 系统的统计机器翻译引擎的构建与部署技术，下文将对构建完毕的机器翻译引擎进行评估，考察引擎的实用性和应用价值。

4 机器翻译引擎的评估

本节从机器翻译的可用性、翻译质量和翻译效率三个方面开展机器翻译引擎的评估。

4.1 机器翻译引擎可用性评估

在机器翻译引擎构建完毕后，本文得到的语言模型、翻译模型与调序模型三个文件的大小之和约为 4.67G，考虑到 Ubuntu 本身运行也需要一定数量的内存，这里把内存设置为 6G，在开启机器翻译引擎服务后，得到了如图 3 所示的结果。

从图 3 中可以看出，机器翻译引擎服务已经成功启动，并在本机的 8090 端口保持监听状态。可以认为，本文所构建的机器翻译引擎能够通过最低限度的硬件测试，已经成功构建和部署，并具备提供翻译服务的可能。

4.2 机器翻译引擎质量评估

构建机器翻译引擎的最终目的在于提供高质量的翻译译文。对翻译结果的评估，目前国际上广泛应用的

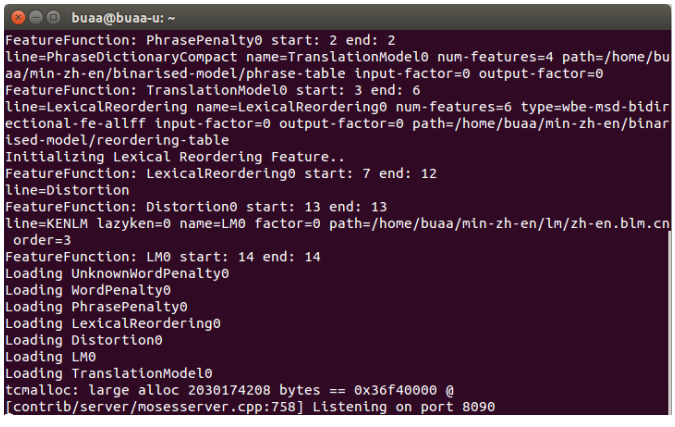

图 3 机器翻译引擎服务启动结果

评价标准是 BLEU 方法[20]。其主要思想是：给定一个标准的译文和自动生成的译文，通过衡量两者间的 n 元匹配程度来对翻译结果进行评价。此外还有一些其他的机器翻译结果评价方式，如 TER[21]、METEOR[22]、RIBES[23]等。限于本文所做工作为面向真实场景应用，需要综合考虑真实场景中的各个因素，例如没有可供参考的标准译文，也并不是实验场景中取得高分的因素都适用于生产环境，因此采取了目标用户评价方式来进行翻译引擎质量的评估。本文并以百度在线翻译、有道在线翻译、微软在线翻译为参考，随机抽取了 100 个句子参考测试，人工观察各个翻译引擎给出的翻译结果。限于篇幅，表 1 中仅罗列了 5 个句子的不同翻译结果：

表 1 五个句子在不同机器翻译引擎中的结果

句 1	北京市 资助 100 家 居家 养老 服务 单位 。
本机	Beijing to support 100 care service unit.
百度	Beijing 100 home care service units.
有道	Beijing grants 100 home endowment service unit.
微软	Beijing-funded 100 home aged care units.
句 2	2015 年 研究 数字 媒体 最 棒 的 10 篇 文章 。
本机	By the study of digital media best 10 articles.
百度	2015 research on the 10 best of the digital media.
有道	A 2015 study the best article 10 digital media.
微软	2015 study by digital media best of 10 articles.
句 3	办法 强调 ，考生 个人 信息 应 当 受到 保护 。
本机	Method stresses, candidates personal information shall be protected.
百度	Approach stressed that candidates personal information should be protected.
有道	To emphasize, the examinee personal information should be protected when.
微软	Approach emphasized that candidates ' personal information should be protected.
句 4	常 吃 四 种 养生 食物 对 口腔 问题 非常 有 好处 。
本机	Eat four kinds of oral health food is very good.
百度	Eat four kinds of health food is very good for oral problems.
有道	Often eat four For oral health food problem is very good.
微软	Eat four kinds of health food that dental problems are very good.
句 5	深度 学习 技术 在 机器 翻译 领域 开始 得到 应用 。
本机	Depth studies in machine translation beginning to be used.
百度	Deep learning technology began to be applied in the field of machine translation.
有道	Deep learning technology began to get application in the field of machine translation.
微软	Advanced learning technology in the field of machine translation began to be applied.

在实验结束后，通过对结果总体分析，我们发现：

(1)在长句子翻译中，本文构建的机器翻译引擎表现不够好，有一些句子出现了明显的错误，如漏译、错译，或者当语料无法覆盖时，多次出现 UNK(未知)的现

象；(2)在相对较短句子的翻译中，本文构建的翻译引擎在总体上要好于百度在线机器翻译引擎，质量上与有道、微软在线机器翻译结果相当，但在译文流畅性上要明显弱于后两者；(3)在翻译质量的稳定性上，与百度、有道、微软的在线机器翻译相比，还相对较弱。

通过深入的分析，我们发现导致上述结果的原因主要有：

(1)本文构建的机器翻译引擎在训练翻译模型时使用的语料在规模上依然相对较小，无法完全覆盖所有词的所有可能翻译，因而带来了部分单词在翻译过程中的错漏现象；

(2)输入语句的中文分词错误会加剧翻译结果的错误，同时中文词汇总量巨大也弱化了翻译模型训练的准确度；这两个因素会降低最终翻译结果的总体质量，具体表现是各个翻译引擎得到的英文译文质量从总体上看均不够好；

(3)在语言模型训练时，本文采用的是训练翻译模型时的译文语料，无论是质量还是规模上都还较为欠缺，造成了译文调序不佳的问题；

(4)然而，本系统在语料规模较小、计算资源匮乏的条件下，与百度、微软的商业翻译引擎相比，总体上依然能够取到较好的结果，尽管与有道相比稍有欠佳，然质量相差并不大。总的来说，可以认为本文研制的中英机器翻译引擎达到了本领域研究的较好水平。

值得指出的是，包括本文研制的机器翻译引擎在内，目前所有的中英机器翻译引擎译文质量都还不够理想，亟待新技术的引入以及语料规模的扩大来促进中外方向机器翻译引擎技术的发展。

4.3 机器翻译引擎翻译效率评估

在实验过程中，我们记录了每一个句子的翻译耗时，最后得到 100 个句子的平均耗时约为 3.51 秒，其中一个句子的翻译耗时如图 4 所示：

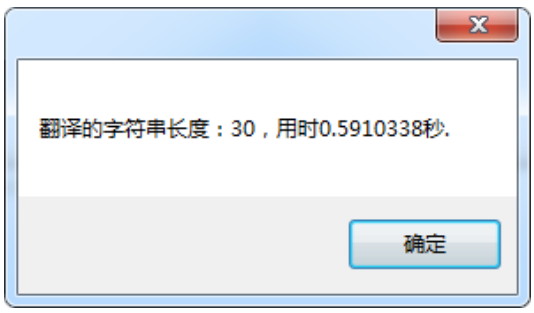

图 4 机器翻译引擎翻译耗时示意图

尽管一些句子的翻译耗时较长，但在整个实验过程中，翻译引擎并没有出现崩溃或者停止服务的情况。

从总体上来看，囿于计算资源不足，本文所构建的机器翻译引擎翻译效率还不够高，但值得肯定的是，系统能够在较为严苛的硬件条件下依然保持较好的鲁棒性。可以预见，当计算资源提升后，机器翻译引擎的翻译效率也会随之提高。

5 结语

本文首先对机器翻译引擎的构建思路进行了全面的规划，包括简要介绍了 Moses 系统及其特性、理清了引擎的构建思路、形成了引擎构建的总体规划；在此基础上把机器翻译引擎的构建与部署划分为四个关键阶段进行了详细的描述，包括系统环境与语料的准备、三个核心模型的构建、引擎多个参数的调优与系统的整体优化和引擎部署时硬件资源的预估和部署方式；然后基于大规模的中英语料构建了一个真实可用的机器翻译引擎，最后从机器翻译引擎的可用性、翻译结果质量、翻译效率三个方面对本文构建的引擎进行了详细的评估，其中在翻译质量评估时，与百度、有道、微软等在线机器翻译服务提供商进行了实验对比测试，并给出了可能导致实验结果的原因。

基于实验结果和对结果的分析，下一步，我们将在采用最新技术的同时，进一步扩大语料规模、提高硬件计算资源，从而解决实验过程中出现的一些不足，进而提升机器翻译的质量和效率。

参考文献

[1] Koehn P. Statistical machine translation[M]. Cambridge University Press, 2009.

[2] Koehn P, Och F J, Marcu D. Statistical phrase-based translation[C]//Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1. Association for Computational Linguistics, 2003: 48-54.

[3] Brown P F, Pietra V J D, Pietra S A D, et al. The mathematics of statistical machine translation: Parameter estimation[J]. Computational linguistics, 1993, 19(2): 263-311.

[4] Koehn P, Hoang H, Birch A, et al. Moses: Open source toolkit for statistical machine translation[C]//Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions. Association for Computational Linguistics, 2007: 177-180.

[5] Li Z, Callison-Burch C, Dyer C, et al. Joshua: An open source toolkit for parsing-based machine translation[C]//Proceedings of the Fourth Workshop on Statistical Machine Translation.

Association for Computational Linguistics, 2009: 135-139.

[6] 黄书剑.统计机器翻译中的词对齐研究[D]. 江苏：南京大学，2012.

[7] Heafield K. KenLM: Faster and smaller language model queries[C]//Proceedings of the Sixth Workshop on Statistical Machine Translation. Association for Computational Linguistics, 2011: 187-197.

[8] Wuebker J, Ney H, Zens R. Fast and scalable decoding with language model look-ahead for phrase-based statistical machine translation[C]//Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2. Association for Computational Linguistics, 2012: 28-32.

[9] Chen B, Foster G F, Kuhn R. Adaptation of Reordering Models for Statistical Machine Translation[C]//HLT-NAACL. 2013: 938-946.

[10] Li J, Marton Y, Resnik P, et al. A unified model for soft linguistic reordering constraints in statistical machine translation[C]//Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics. 2014, 1: 1123-1133.

[11] Cherry C, Foster G. Batch tuning strategies for statistical machine translation[C]//Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, 2012: 427-436.

[12] Chen B, Guo H, Kuhn R. Multi-level Evaluation for Machine Translation[J]. EMNLP 2015, 2015: 361.

[13] Callison-Burch C, Koehn P, Monz C, et al. Findings of the 2011 workshop on statistical machine translation[C]//Proceed ings of the Sixth Workshop on Statistical Machine Translation. Association for Computational Linguistics, 2011: 22-64.

[14] Koehn P, Hoang H. Factored Translation Models[C]//EMNLP-CoNLL. 2007: 868-876.

[15] Federico M, Bertoldi N, Cettolo M. IRSTLM: an open source toolkit for handling large scale language models[C]//Interspeech. 2008: 1618-1621.

[16] Casacuberta F, Vidal E. GIZA++: Training of statistical translation models[J]. 2007.

[17] Gao Q, Vogel S. Parallel implementations of word alignment tool[C]//Software Engineering, Testing, and Quality Assurance for Natural Language Processing. Association for Computational Linguistics, 2008: 49-57.

[18] 杨南. 基于神经网络学习的统计机器翻译研究[D].合肥:中国科学技术大学,2014.

[19] Vaswani A, Zhao Y, Fossum V, et al. Decoding with Large-Scale Neural Language Models Improves Translation. Proceedings of Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, Washington, USA: Association for Computational Linguistics, 2013. 1387-1392.

[20] Papineni K, Roukos S, Ward T, et al. BLEU: a method for automatic evaluation of machine translation[C]//Proceedings of the 40th annual meeting on association for computational linguistics. Association for Computational Linguistics, 2002: 311-318.

[21] Och F J. Minimum error rate training in statistical machine translation[C]//Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1. Association for Computational Linguistics, 2003: 160-167.

[22] Banerjee S, Lavie A. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments[C]//Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization. 2005, 29: 65-72.

[23] Isozaki H, Hirao T, Duh K, et al. Automatic evaluation of translation quality for distant language pairs[C]//Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2010: 944-952.