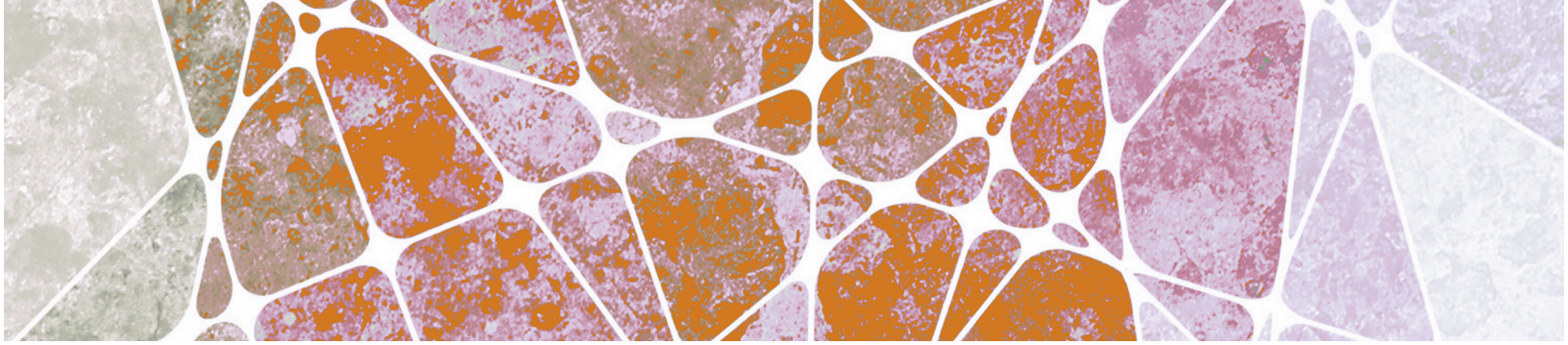
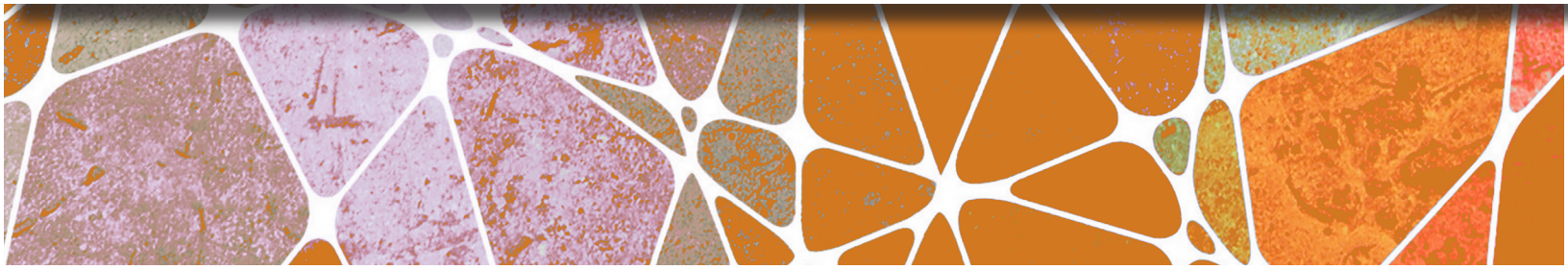
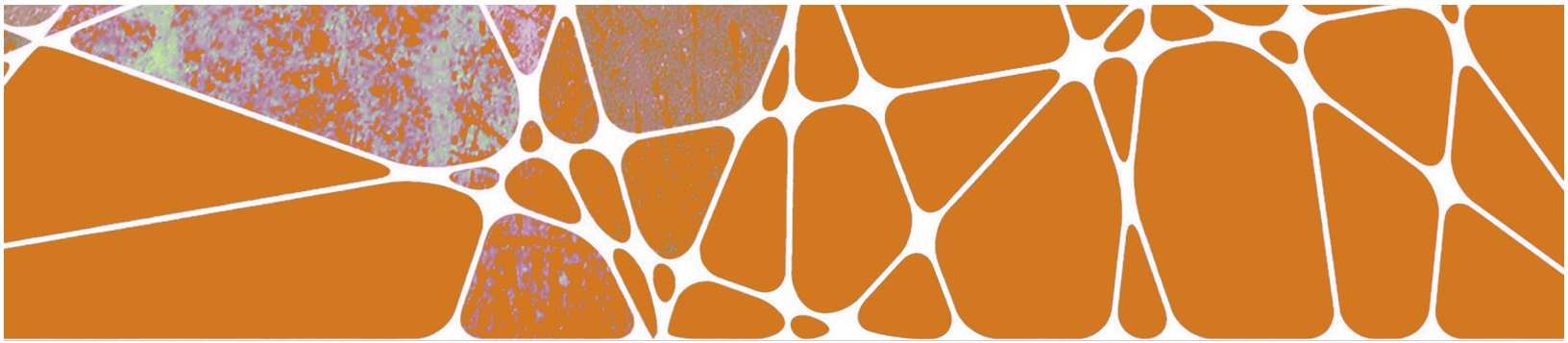


SYSTEMS THAT LEARN @ CSAIL



WELCOME PACKET



SYSTEMS THAT LEARN @ CSAIL

MIT CSAIL ALLIANCES

Massachusetts Institute of Technology

MESSAGE from the EXECUTIVE DIRECTOR

Welcome to SystemsThatLearn@CSAIL!

SystemsThatLearn@CSAIL (STL) is MIT's AI research initiative in collaboration with industry. As a member of this initiative you will have a voice in shaping the research, access to the most cutting edge technologies developed in the space, close collaboration with world-renowned researchers and connections to top student talent.

Each member has 1 board position on the SystemsThatLearn@CSAIL governing board. The board consists of member companies, researchers, the faculty directors and the executive director. The board meets and frames problem statements on issues relevant to our members. The problem statements are then crafted into an RPF which is sent to the entire lab (117 researchers). The proposals are then gathered and reviewed by the member company representatives and voted on for funding allocation. Details of the structure of the executive board are included in this packet.

More broadly, SystemsThatLearn@CSAIL is a research initiative under the management of CSAIL Alliances, the division of CSAIL that manages industry and organizational collaborations. CSAIL Alliances also manages the CyberSecurity@CSAIL research initiative, the CSAIL Alliance Program, the Visiting Industry researcher program, student recruiting activities, CSAIL Start-up Connect and professional development programs. As part of SystemsThatLearn@CSAIL you have benefits in each of these areas. Details are included in the Benefits Grid in this packet.

We hope you find this information useful. If you have any questions, I would be happy to answer them.

This year our general Alliances Annual Meeting will take place June 5-7, 2017, at MIT. This event will showcase the latest research from across the lab, as well as demonstrate some of our emerging technologies. You will also have an opportunity to meet some of our start-ups coming out of the lab, as well as meet CSAIL students at the poster reception. This event is a great opportunity to connect with the staff, students and research at the lab. The Annual Meeting site is currently available and open for registration: cap.csail.mit.edu/annualmeeting. You must be logged into the CSAIL Alliances website to access the page. Admission is part of your membership in STL@CSAIL and Alliances and there is no limit to the number of people your company can send. We hope you will be able to attend.

Also, this year we are taking part in the second annual Cambridge 2 Cambridge (cambridge2cambridge.csail.mit.edu) international cybersecurity challenge with the University of Cambridge in the UK. This event consists of a hackathon that will bring together top students from the US and UK. The event will take place at the University of Cambridge on July 24-27, 2017. The hackathon is an open event and we encourage you to attend.

If you have any questions or comments, please do not hesitate to contact me.

Sincerely,

Lori Glover

Executive Director, SystemsThatLearn@CSAIL
Managing Director, MIT CSAIL Alliances

MEMBER MODEL and BENEFITS

LEVERAGED RESOURCES: EXISTING PROJECTS

Member companies will have access to research and testing opportunities in several current non-industry sponsored projects at CSAIL, as well as provide guidance for new seed projects as part of the collaborative initiative.

THE MEMBERSHIP MODEL:

Through initiatives, member companies engage in close interactions with researchers and students in the space. Member companies will have the opportunity to interact with multiple research projects that span the full spectrum of machine learning/artificial intelligence and analytics. We will collaborate closely with industry to provide real-world applications and drive impact. Our team of world-class researchers covers the full spectrum of research in systems and machine learning.

Systems That Learn @ CSAIL industry partners will:

- Participate in the Systems That Learn @ CSAIL Advisory Board. Each member company will have one (1) representative on the board. The board will advise the initiative on industry needs, provide feedback on existing research and advise future research direction through seeded projects. This board will help shape the priorities of the initiative.
- Access in-depth exploration of CSAIL research in AI, machine learning and data analytics. As part of this initiative, we will leverage the work of 15+ existing research projects. Members will have unprecedented access to the research and the research teams.
- Test application of tools developed to real-world situations and explore new projects.
- Access tools created as part of the initiative via MIT open source license through the CSAIL Technology Application Portal.
- Participate in in-depth interactions and shared learning on topics of particular interest to each company. Close interaction with the researchers engaged in what matters most to your company.
- Members will be invited to attend one (1) annual meeting per year and may send up to 10 representatives to enable broad exposure to teams who are working on these issues. Members will also be invited to participate in the Systems That Learn @ CSAIL lecture series and workshops held throughout the year.
- Access additional research groups, researchers, and students within MIT's Computer Science and Artificial Intelligence Laboratory through CSAIL's Alliance Program (CAP) at the Affiliate level. Details include:
- Access to the lab-wide annual 2-day member only Annual Meeting held in May/June each year in addition to SystemsThatLearn@CSAIL Annual Meeting.
- Connect with the latest CSAIL research from across the groups: Big Data, Wireless, Robotics, HCI, Computer Vision, Security/Crypto, Natural Language, Computational Biology, Algorithms, Architecture, Theory, Artificial Intelligence and Machine Learning.
- Access technical talks from our world-renowned researchers and visiting researchers held on campus each month both live and virtually.
- A "Student Profile Book" containing resumes and research summaries of current CSAIL students published each year.
- Advertise open position announcements within CSAIL

(continued)

THE MEMBERSHIP MODEL (cont.):

Systems That Learn @ CSAIL industry partners will be able to:

- Have two tech talks/info sessions per academic year.
- Have your ompany logo included on CSAIL Alliance Program site and in conference material
- Have members of your company are welcome to visit CSAIL for a private lab visit, tour, demos and meeting with faculty/researchers (1 per year).
- Network with faculty, students and other members at networking events throughout the year.
- Fully access the member-only site with search function: papers, student projects, resume book, research, tech talk and seminar videos, demos, conference slides, business use cases and more
- Obtain a 10% discount on professional education classes through MIT school of Engineering.
- Obtain a 15% discount on open enrollment executive education courses with MIT Sloan School of Management.
- Receive research briefings/ and/or research summaries highlighting the latest CSAIL research 3-4 times per year

OPTIONAL ADDITIONAL ENGAGEMENTS:

In addition to accessing the existing research and shared learning on specific topics of interest, each member company may also enter into company–specific activities such as:

Sponsored Research

If a member company becomes interested in a particular research project and wants to sponsor future development of that project, we can work with members to scope the project and additional funding required. All sponsored research is handled through MIT's Office of Sponsored Programs.

Visiting Industry Researcher (with or without sponsored research)

Member companies will be able to leverage the CSAIL Visiting Industry Researcher Program to embed a researcher within a specific research group. The visiting researcher remains an employee of the member-company and works alongside the researchers and students in a specific area at CSAIL. In addition, the Visiting Industry Researcher is connected to CSAIL/MIT with a customized schedule of lectures, workshops, classes, meetings and events. The CSAIL Alliance Program coordinates this effort and meets monthly with all Industry Visitors.

Consulting

Researchers may be available for consulting opportunities with our industry partners. Consulting agreements are arranged between the researcher and the member-company directly.

Technology Licensing

Member companies interested in licensing developed software and patentable inventions may work with our MIT Technology Licensing Office (TLO) for licensing agreements and options.

INTELLECTUAL PROPERTY:

The goal of SystemsThatLearn @CSAIL is to conduct fundamental research that will significantly impact the field of artificial intelligence and machine learning over the next decade and beyond.

Publication

The majority of the research results will be broadly disseminated via publication. We realize that confidentiality is a particular concern to our partners and we have standard practices to ensure no partner's proprietary information will be released in any publication. Papers generated from this research initiative will be available to our partners BEFORE any publication and in parallel with conference submission.

Open-Source and licensing

Members will be able to utilize software tools developed through this initiative for research testing internally. Additionally, we anticipate that some of the software tools developed will be released as open source and available for use by member companies via MIT's open source license.

Member Intellectual Property

All pre-existing IP owned by the member coming into this initiative will remain the member's intellectual property.

Creation of Joint Intellectual property

If member representatives work with MIT researchers on projects to invent and/or author inventions and software, US laws and rules with regard to joint-ownership of patents and copyrights will be applied. Please note, however, if members make significant use of MIT resources, funds, and/or facilities or invent in area outside the scope of initiative projects, their IP rights will be assigned to MIT.

SYSTEMSTHATLEARN@CSAIL GOVERNING BOARD

Steve Whittaker
Head of Strategic US Academic Research Partnership
British Telecommunications PLC (BT)

Hernán Asorey
Senior Vice President, Product Data Science
Salesforce.com

Surajit Chaudhuri
Distinguished Scientist
Microsoft Corporation

Suhail Shergill
Director, Data Science
The Bank of Nova Scotia

Christopher White
Research Lab Leader
Nokia Bell Labs

Smaine Zeroug
Research Director
Schlumberger-Doll Research

Lori Glover
Executive Director
SystemsThatLearn@CSAIL

Sam Madden
Faculty Co-Director
SystemsThatLearn@CSAIL

Tommi Jaakkola
Faculty Co-Director
SystemsThatLearn@CSAIL

Each member company will have one representative on the board. The board will advise the initiative on industry needs, provide feedback on existing research and advise future research direction through seeded projects. This board will help shape the priorities of the initiative.

SYSTEMSTHATLEARN@CSAIL RESEARCHERS

Access in-depth exploration of CSAIL research in AI, machine learning and data analytics. As part of this initiative, we will leverage the work of 15+ existing research projects. Members will have unprecedented access to the research and the research teams.

FACULTY DIRECTORS

SAM MADDEN

Current Research Topics:
Database systems, including main memory databases, data warehousing/analytics, database-as-a-service, and querying data streams and networks of distributed devices such as wireless sensor networks

TOMMI JAAKKOLA

Current Research Topics:
Problems in natural language processing, computational biology (e.g., regulatory models), recommender and other large scale inference problems, as well as information retrieval.

FACULTY

MOHAMMAD ALIZADEH

Current Research Topics:
Building systems that incorporate learning in their control decisions (e.g., for resource management, scheduling, etc.)

SAMAN AMARASINGHE

Current Research Topics:
Compiler optimizations, computer architectures, software engineering and parallel computing

ARVIND

Current Research Topics:
Synthesis and verification of large digital systems described using Guarded Atomic Actions; and Memory Models and Cache Coherence Protocols for parallel architectures and languages

HARI BALAKRISHNAN

Current Research Topics:
Centralized data planes for datacenters and enterprise networks. Programmable data planes for routers

REGINA BARZILAY

Current Research Topics:
Natural language processing- medical data focus

SYSTEMSTHATLEARN@CSAIL RESEARCHERS

FACULTY (cont.)

BONNIE BERGER

Current Research Topics:
Deep learning algorithms for personalized medical diagnostics and integration of heterogeneous big biological data

TAMARA BRODERICK

Current Research Topics:
Bayesian inference and graphical models—with an emphasis on scalable, nonparametric, and unsupervised learning.

SRINIVAS DEVADAS

Current Research Topics:
Computer architecture and computer security

ALAN EDELMAN

Current Research Topics:
Proving “pure math” theorems in random matrix theory, developing numerical algorithms, and improving software for high performance computing

JOHN FISHER

Current Research Topics:
Signal level approaches to multi-modal data fusion, distributed inference under resource constraints, resource management in sensor networks, and analysis of seismic and radar images

WILLIAM FREEMAN

Current Research Topics:
Motion re-rendering, computational photography, and learning for vision

JIM GLASS

Current Research Topics:
Machine learning, neural models

POLINA GOLLAND

Current Research Topics:
Building systems that enable medical image analysis and visualization

AMAR GUPTA

Current Research Topics:
Digital Health from technical, business, legal, entrepreneurial, and public policy aspects

SYSTEMSTHATLEARN@CSAIL RESEARCHERS

FACULTY (cont.)

JOHN GUTTAG

Current Research Topics:
Application of advanced computational techniques to medicine

PIOTR INDYK

Current Research Topics:
Developing algorithms that work well when deployed in large scale systems, focusing on algorithms that parallelize well, have predictable or interpretable performance

STEFANIE JEGELKA

Current Research Topics:
Algorithmic machine learning, modeling, optimization algorithms, theory and applications - the mathematical structure for discrete and combinatorial machine learning problems

LALANA KAGAL

Current Research Topics:
Modeling how social norms and legal rules work in society in order to automate the compliance of policy in information systems

BORIS KATZ

Current Research Topics:
Natural language understanding and generation as well as multimodal information access, knowledge representation, human computer interaction, and event recognition

ANDREW LO

Current Research Topics:
Computational finance

NANCY ANN LYNCH

Current Research Topics:
Feedback-based algorithms that allow individual agents to adjust their behavior in response to observations about the environment

ALEKSANDER MADRY

Current Research Topics:
Algorithmic graph theory, i.e., design and analysis of very efficient (approximation) algorithms for fundamental graph problems

WOJCIECH MATUSIK

Current Research Topics:
Direct digital manufacturing and computer graphics

SYSTEMSTHATLEARN@CSAIL RESEARCHERS

FACULTY (cont.)

AUDE OLIVA

Current Research Topics:
Combining methods from computer science, neuroscience and cognitive science to explain and model how perception and cognition are realized in human and machine

ALEX “SANDY” PENTLAND

Current Research Topics:
Computational social science, big data, privacy, and wearable computing

NICHOLAS ROY

Current Research Topics:
Building unmanned vehicles that can fly without GPS through unmapped indoor environments, robots that can drive through unmapped cities, and to build social robots that can quickly learn what people want without being annoying or intrusive

RONITT RUBINFELD

Current Research Topics:
Randomized algorithms, sublinear time algorithms, property testing, program checking, and learning theory

DANIEL SANCHEZ

Current Research Topics:
Performance guarantees in shared clusters and reconfigurable, adaptive memory systems

NIR SHAVIT

Current Research Topics:
High-performance environments for running ML on multicores

ARMANDO SOLAR-LEZAMA

Current Research Topics:
Techniques and tools that exploit automated reasoning and large amounts of computing power to tackle challenging programming problems

PETER SZOLIVITZ

Current Research Topics:
The application of AI methods to problems of medical decision making, natural language processing to extract meaningful data from clinical narratives to support translational medicine, and the design of information systems for health care institutions and patients

JUSTIN SOLOMON

Current Research Topics:
Geometry/numerics for dealing with 3D data

FACULTY (cont.)

MICHAEL STONEBRAKER

Current Research Topics:
Data base technology, operating systems and the architecture of system software services

RUSSELL TEDRAKE

Current Research Topics:
Control solutions for interesting (underactuated, stochastic, and/or difficult to model) dynamical systems

ANTONIO TORRALBA

Current Research Topics:
Computer vision, machine learning and human visual perception

For addition information about MIT CSAIL PI's, please visit: csail.mit.edu/directory

Computer Science and Artificial Intelligence Lab
Principal Investigators

Scott Aaronson
Visiting Professor
Office:
+1 (617) 324-8356
aaronson@csail.mit.edu

Hal Abelson
Professor
Office:32-G516
+1 (617) 253-5856
hal@mit.edu

Ted Adelson
Professor
Office:32-D428
+1 (617) 253-0645
adelson@csail.mit.edu

Anant Agarwal
Professor
Office:
+1 (617) 253-1448
agarwal@edx.org

Mohammad Alizadeh
Assistant Professor
Office:32-G920

alizadeh@mit.edu

Saman Amarasinghe
Professor
Office:32-G744
+1 (617) 253-8879
saman@csail.mit.edu

Arvind
Professor
Office:32-G866
+1 (617) 253-6090
arvind@csail.mit.edu

Hari Balakrishnan
Professor
Office:32-G940
+1 (617) 253-8713
hari@csail.mit.edu

Regina Barzilay
Professor
Office:32-G468
+1 (617) 258-5706
regina@csail.mit.edu

Bonnie Berger
Professor
Office:32-G574
+1 (617) 253-1827
bab@csail.mit.edu

Tim Berners-Lee
Professor
Office:32-G524
+1 (617) 253-5702
timbl@w3.org

Judy Brewer
Principal Research...
Office:32-385
+1 (617) 258-9741
jbrewer@w3.org

Tamara Broderick
Assistant Professor
Office:32-G498
+1 (617) 324-6749
tbroderick@csail.mit.edu

Rodney Brooks
Professor Emeritus
Office:

brooks@csail.mit.edu

Michael Carbin
Assistant Professor
Office:32-G782
+1 (617) 253-5881
mcarbin@csail.mit.edu

Adam Chlipala
Associate Professor
Office:32-G842
+1 (617) 324-8439
adamc@csail.mit.edu

David Clark
Senior Research Scientist...
Office:32-G816
+1 (617) 253-6003
ddc@csail.mit.edu

F. J. Corbato
Professor Emeritus
Office:
+1 (617) 253-6001
corbato@mit.edu

Costis Daskalakis
Associate Professor
Office:32-G694
+1 (617) 253-9643
costis@csail.mit.edu

Randall Davis
Professor
Office:32-237
+1 (617) 253-5879
davis@csail.mit.edu

Erik Demaine
Professor
Office:32-G680
+1 (617) 253-6871
edemaine@mit.edu

Jack Dennis
Professor Emeritus
Office:32-G868
+1 (617) 253-6856
dennis@csail.mit.edu

Srini Devadas
Professor
Office:32-G844
+1 (617) 253-0454
devadas@csail.mit.edu

Fredo Durand
Professor
Office:32-D424
+1 (617) 253-7223
fredo@csail.mit.edu

Alan Edelman
Professor
Office:32-G780
+1 (617) 253-1355
edelman@csail.mit.edu

Joel Emer
Professor of the Practice...
Office:32-G864
+1 (617) 258-9190
emer@csail.mit.edu

John Fisher
Senior Research Scientist...
Office:32-D468
+1 (617) 253-0788
fisher@csail.mit.edu

William Freeman
Professor
Office:32-D476
+1 (617) 253-8828
billf@csail.mit.edu

Dave Gifford
Professor
Office:32-G542
+1 (617) 253-6039
gifford@csail.mit.edu

Jim Glass
Senior Research Scientist...
Office:32-G444
+1 (617) 253-1640
glass@mit.edu

Michael Goemans
Professor
Office:32-G618
+1 (617) 253-2688
goemans@csail.mit.edu

Shafi Goldwasser
Professor
Office:32-G682
+1 (617) 253-5914
shafi@csail.mit.edu

Polina Golland
Professor
Office:32-D470
+1 (617) 253-8005
polina@csail.mit.edu

Eric Grimson
Chancellor for Academic...
Office:3-221
+1 (617) 253-5415
welg@mit.edu

John Guttag
Professor
Office:32-G966
+1 (617) 253-6022
guttag@csail.mit.edu

D. Fox Harrell
Associate Professor
Office:14N-207
+1 (617) 324-4278
fox@csail.mit.edu

Berthold Horn
Professor
Office:32-D434
+1 (617) 253-5863
bkph@csail.mit.edu

Piotr Indyk
Professor
Office:32-G642
+1 (617) 452-3402
indyk@csail.mit.edu

Tommi Jaakkola
Professor
Office:32-G470
+1 (617) 253-0440
tommi@csail.mit.edu

Daniel Jackson
Professor
Office:32-G704
+1 (617) 258-8471
dnj@csail.mit.edu

Jeffrey Jaffe
Principal Research...
Office:32-386A
+1 (617) 253-7697
jeff@w3.org

Stefanie Jegelka
Assistant Professor
Office:32-G472

stefje@mit.edu

Frans Kaashoek
Professor
Office:32-G992
+1 (617) 253-7149
kaashoek@csail.mit.edu

Leslie Kaelbling
Professor
Office:32-G486
+1 (617) 258-9695
lpk@csail.mit.edu

Lalana Kagal
Principal Research...
Office:32-G518
+1 (617) 253-5845
lkagal@csail.mit.edu

David Karger
Professor
Office:32-G592
+1 (617) 258-6167
karger@csail.mit.edu

Dina Katabi
Professor
Office:32-G936
+1 (617) 324-6027
dina@csail.mit.edu

Boris Katz
Principal Research...
Office:32-G430
+1 (617) 253-6032
boris@csail.mit.edu

Computer Science and Artificial Intelligence Lab
Principal Investigators

Manolis Kellis
Professor
Office:32-D524
+1 (617) 253-2419
manoli@mit.edu

Jonathan Kelner
Associate Professor
Office:32-G618
+1 (617) 253-4344
kelner@mit.edu

Butler Lampson
Adjunct Professor
Office:32-G924
+1 (617) 253-6004
blampson@microsoft.com

Tom Leighton
Professor
Office:32-G594
+1 (617) 253-5876
ftl@csail.mit.edu

Charles Leiserson
Professor
Office:32-G768
+1 (617) 253-5833
cel@csail.mit.edu

John Leonard
Professor
Office:32-232
+1 (617) 253-5305
jleonard@csail.mit.edu

Barbara Liskov
Institute Professor
Office:32-G942
+1 (617) 253-5886
liskov@csail.mit.edu

Andrew Lo
Professor
Office:
+1 (617) 253-0920
alo@mit.edu

Bill Long
Research Affiliate
Office:32-256
+1 (617) 253-3508
wjl@mit.edu

Tomas Lozano-Perez
Professor
Office:32-G492
+1 (617) 253-7889
tlp@csail.mit.edu

Nancy Lynch
Professor
Office:32-G668
+1 (617) 253-7225
lynch@csail.mit.edu

Sam Madden
Professor
Office:32-G938
+1 (617) 258-6643
madden@csail.mit.edu

Aleksander Madry
Assistant Professor
Office:32-G666
+1 (617) 324-6739
madry@mit.edu

Thomas Magnanti
Institute Professor
Office:32-D784
+1 (617) 253-6604
magnanti@csail.mit.edu

Wojciech Matusik
Associate Professor
Office:
+1 (617) 324-8432
wojciech@csail.mit.edu

Albert R. Meyer
Professor
Office:32-G624
+1 (617) 253-6024
meyer@csail.mit.edu

Silvio Micali
Professor
Office:32-G644
+1 (617) 253-5949
silvio@csail.mit.edu

Rob Miller
Professor
Office:32-G718
+1 (617) 324-6028
rcm@csail.mit.edu

Ankur Moitra
Assistant Professor
Office:32-G594
+1 (617) 253-5876
moitra@csail.mit.edu

Robert Morris
Professor
Office:32-G972
+1 (617) 253-5983
rtm@csail.mit.edu

Joel Moses
Institute Professor
Office:32-249
+1 (617) 253-8592
moses@csail.mit.edu

Dana Moshkovitz
Visiting Professor
Office:

dmoshkov@csail.mit.edu

Stefanie Mueller
Associate Professor
Office:32-214

stefanie.mueller@mit.edu

Una-May O'Reilly
Principal Research...
Office:32-D534
+1 (617) 253-6437
unamay@csail.mit.edu

Aude Oliva
Principal Research...
Office:32-D432
+1 (617) 452-2492
oliva@mit.edu

Li Shiuan Peh
Visiting Professor
Office:
+1 (617) 324-8428
peh@csail.mit.edu

Tomaso Poggio
Professor
Office:46-5177
+1 (617) 258-9501
tp@csail.mit.edu

Martin Rinard
Professor
Office:32-G828
+1 (617) 258-6922
rinard@csail.mit.edu

Ronald Rivest
Institute Professor
Office:32-G692
+1 (617) 253-5880
rivest@mit.edu

Ruth Rosenholtz
Principal Research...
Office:32-D532
+1 (617) 324-0269
rruth@csail.mit.edu

Nicholas Roy
Associate Professor
Office:32-330
+1 (617) 253-2517
nickroy@csail.mit.edu

Ronitt Rubinfeld
Professor
Office:32-G698
+1 (617) 253-0884
ronitt@csail.mit.edu

Daniela Rus
Director
Office:32-368
+1 (617) 258-7567
rus@csail.mit.edu

Jerry Saltzer
Professor Emeritus
Office:
+1 (617) 253-6016
saltzer@mit.edu

Daniel Sanchez
Assistant Professor
Office:32-G838
+1 (617) 715-4886
sanchez@csail.mit.edu

Stephanie Seneff
Senior Research Scientist...
Office:32-G438
+1 (617) 253-0451
seneff@csail.mit.edu

Julie Shah
Assistant Professor
Office:32-332
+1 (617) 324-4879
julie_a_shah@csail.mit.edu

Nir Shavit
Professor
Office:32-G622
+1 (617) 324-8440
shanir@csail.mit.edu

Peter Shor
Professor
Office:32-G574
+1 (617) 253-1827
shor@csail.mit.edu

Howard Shrobe
Principal Research...
Office:32-225
+1 (617) 253-7877
hes@csail.mit.edu

Michael Sipser
Professor
Office:32-G594
+1 (617) 253-4992
sipser@MIT.EDU

Armando Solar-Lezama
Associate Professor
Office:32-G840
+1 (617) 258-9727
asolar@csail.mit.edu

Karen Sollins
Principal Research...
Office:32-G818
+1 (617) 253-6006
sollins@csail.mit.edu

Justin Solomon
Assistant Professor
Office:32-D460

jsolomon@mit.edu

David Sontag
Assistant Professor
Office:32-G464

dsontag@csail.mit.edu

Michael Stonebraker
Adjunct Professor
Office:32-G922
+1 (617) 253-3538
stonebraker@csail.mit.edu

Gerald Sussman
Professor
Office:32-G508
+1 (617) 253-5874
gjs@csail.mit.edu

Peter Szolovits
Professor
Office:32-254
+1 (617) 253-3476
psz@mit.edu

Computer Science and Artificial Intelligence Lab
Principal Investigators

Russ Tedrake
Professor
Office:32-380B
+1 (617) 253-1778
russt@csail.mit.edu

Joshua Tenenbaum
Professor
Office:46-4015
+1 (617) 452-2010
jbt@csail.mit.edu

Chris Terman
Senior Lecturer
Office:32-G790
+1 (617) 253-6038
cjt@mit.edu

Bruce Tidor
Professor
Office:32-212
+1 (617) 253-7258
tidor@mit.edu

Antonio Torralba
Professor
Office:32-D462
+1 (617) 324-0900
torralba@csail.mit.edu

Vinod Vaikuntanathan
Associate Professor
Office:32-G696
+1 (617) 324-8444
vinodv@csail.mit.edu

Steve Ward
Professor
Office:32-G786
+1 (617) 253-6036
ward@mit.edu

Ron Weiss
Professor
Office:NE47-223
+1 (617) 253-8966
rweiss@csail.mit.edu

Daniel Weitzner
Principal Research...
Office:32-G526
+1 (617) 253-8036
djweitzner@csail.mit.edu

Ryan Williams
Associate Professor
Office:32-G638

rrwilliams@gmail.com

Brian Williams
Professor
Office:32-227
+1 (617) 253-2739
williams@csail.mit.edu

Virginia Vassilevska Williams
Associate Professor
Office:32-G640

virgi@mit.edu

Alan Willsky
Professor Emeritus
Office:32-D582
+1 (617) 253-2356
willsky@mit.edu

Patrick Winston
Professor
Office:32-251
+1 (617) 253-6754
phw@csail.mit.edu

Jack Wisdom
Professor
Office:
+1 (617) 253-7730
wisdom@csail.mit.edu

Matei Zaharia
Assistant Professor
Office:
+1 (617) 253-0004
matei@csail.mit.edu

Nickolai Zeldovich
Associate Professor
Office:32-G994
+1 (617) 253-6005
nickolai@csail.mit.edu

Victor Zue
Professor
Office:32-G422
+1 (617) 253-8513
zue@csail.mit.edu

Rationalizing Neural Predictions

Tao Lei, Regina Barzilay and Tommi Jaakkola
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
{taolei, regina, tommi}@csail.mit.edu

Abstract

Prediction without justification has limited applicability. As a remedy, we learn to extract pieces of input text as justifications – rationales – that are tailored to be short and coherent, yet sufficient for making the same prediction. Our approach combines two modular components, generator and encoder, which are trained to operate well together. The generator specifies a distribution over text fragments as candidate rationales and these are passed through the encoder for prediction. Rationales are never given during training. Instead, the model is regularized by desiderata for rationales. We evaluate the approach on multi-aspect sentiment analysis against manually annotated test cases. Our approach outperforms attention-based baseline by a significant margin. We also successfully illustrate the method on the question retrieval task.[1]

1 Introduction

Many recent advances in NLP problems have come from formulating and training expressive and elaborate neural models. This includes models for sentiment classification, parsing, and machine translation among many others. The gains in accuracy have, however, come at the cost of interpretability since complex neural models offer little transparency concerning their inner workings. In many applications, such as medicine, predictions are used to drive critical decisions, including treatment options. It is necessary in such cases to be able to verify and understand the underlying basis for the decisions. Ideally, complex neural models would not only yield improved performance but would also offer interpretable justifications – rationales – for their predictions.

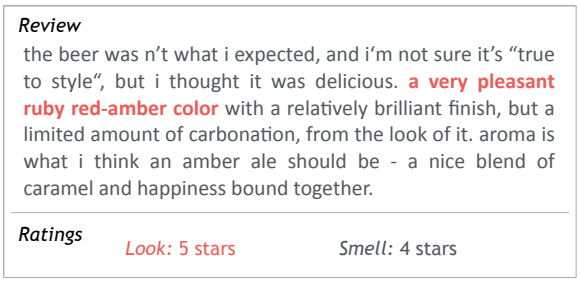


Figure 1: An example of a review with ranking in two categories. The rationale for Look prediction is shown in bold.

In this paper, we propose a novel approach to incorporating rationale generation as an integral part of the overall learning problem. We limit ourselves to extractive (as opposed to abstractive) rationales. From this perspective, our rationales are simply subsets of the words from the input text that satisfy two key properties. First, the selected words represent short and coherent pieces of text (e.g., phrases) and, second, the selected words must alone suffice for prediction as a substitute of the original text. More concretely, consider the task of multi-aspect sentiment analysis. Figure 1 illustrates a product review along with user rating in terms of two categories or aspects. If the model in this case predicts five star rating for color, it should also identify the phrase *"a very pleasant ruby red-amber color"* as the rationale underlying this decision.

In most practical applications, rationale genera-

[1]Our code and data are available at https://github.com/taolei87/rcnn.

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 107–117,
Austin, Texas, November 1-5, 2016.

tion must be learned entirely in an unsupervised manner. We therefore assume that our model with rationales is trained on the same data as the original neural models, without access to additional rationale annotations. In other words, target rationales are never provided during training; the intermediate step of rationale generation is guided only by the two desiderata discussed above. Our model is composed of two modular components that we call the generator and the encoder. Our generator specifies a distribution over possible rationales (extracted text) and the encoder maps any such text to task specific target values. They are trained jointly to minimize a cost function that favors short, concise rationales while enforcing that the rationales alone suffice for accurate prediction.

The notion of what counts as a rationale may be ambiguous in some contexts and the task of selecting rationales may therefore be challenging to evaluate. We focus on two domains where ambiguity is minimal (or can be minimized). The first scenario concerns with multi-aspect sentiment analysis exemplified by the beer review corpus (McAuley et al., 2012). A smaller test set in this corpus identifies, for each aspect, the sentence(s) that relate to this aspect. We can therefore directly evaluate our predictions on the sentence level with the caveat that our model makes selections on a finer level, in terms of words, not complete sentences. The second scenario concerns with the problem of retrieving similar questions. The extracted rationales should capture the main purpose of the questions. We can therefore evaluate the quality of rationales as a compressed proxy for the full text in terms of retrieval performance. Our model achieves high performance on both tasks. For instance, on the sentiment prediction task, our model achieves extraction accuracy of 96%, as compared to 38% and 81% obtained by the bigram SVM and a neural attention baseline.

2 Related Work

Developing sparse interpretable models is of considerable interest to the broader research community(Letham et al., 2015; Kim et al., 2015). The need for interpretability is even more pronounced with recent neural models. Efforts in this area include analyzing and visualizing state activation (Hermans and Schrauwen, 2013; Karpathy et al., 2015; Li et al., 2016), learning sparse interpretable word vectors (Faruqui et al., 2015b), and linking word vectors to semantic lexicons or word properties (Faruqui et al., 2015a; Herbelot and Vecchi, 2015).

Beyond learning to understand or further constrain the network to be directly interpretable, one can estimate interpretable proxies that approximate the network. Examples include extracting "if-then" rules (Thrun, 1995) and decision trees (Craven and Shavlik, 1996) from trained networks. More recently, Ribeiro et al. (2016) propose a model-agnostic framework where the proxy model is learned only for the target sample (and its neighborhood) thus ensuring locally valid approximations. Our work differs from these both in terms of what is meant by an explanation and how they are derived. In our case, an explanation consists of a concise yet sufficient portion of the text where the mechanism of selection is learned jointly with the predictor.

Attention based models offer another means to explicate the inner workings of neural models (Bahdanau et al., 2015; Cheng et al., 2016; Martins and Astudillo, 2016; Chen et al., 2015; Xu and Saenko, 2015; Yang et al., 2015). Such models have been successfully applied to many NLP problems, improving both prediction accuracy as well as visualization and interpretability (Rush et al., 2015; Rocktäschel et al., 2016; Hermann et al., 2015). Xu et al. (2015) introduced a stochastic attention mechanism together with a more standard soft attention on image captioning task. Our rationale extraction can be understood as a type of stochastic attention although architectures and objectives differ. Moreover, we compartmentalize rationale generation from downstream encoding so as to expose knobs to directly control types of rationales that are acceptable, and to facilitate broader modular use in other applications.

Finally, we contrast our work with rationale-based classification (Zaidan et al., 2007; Marshall et al., 2015; Zhang et al., 2016) which seek to improve prediction by relying on richer annotations in the form of human-provided rationales. In our work, rationales are never given during training. The goal is to learn to generate them.

3 Extractive Rationale Generation

We formalize here the task of extractive rationale generation and illustrate it in the context of neural models. To this end, consider a typical NLP task where we are provided with a sequence of words as input, namely $\mathbf{x} = \{x_1, \cdots, x_l\}$, where each $x_t \in \mathbb{R}^d$ denotes the vector representation of the i-th word. The learning problem is to map the input sequence $\mathbf{x}$ to a target vector in $\mathbb{R}^m$. For example, in multi-aspect sentiment analysis each coordinate of the target vector represents the response or rating pertaining to the associated aspect. In text retrieval, on the other hand, the target vectors are used to induce similarity assessments between input sequences. Broadly speaking, we can solve the associated learning problem by estimating a complex parameterized mapping $\mathbf{enc}(\mathbf{x})$ from input sequences to target vectors. We call this mapping an *encoder*. The training signal for these vectors is obtained either directly (e.g., multi-sentiment analysis) or via similarities (e.g., text retrieval). The challenge is that a complex neural encoder $\mathbf{enc}(\mathbf{x})$ reveals little about its internal workings and thus offers little in the way of justification for why a particular prediction was made.

In extractive rationale generation, our goal is to select a subset of the input sequence as a *rationale*. In order for the subset to qualify as a rationale it should satisfy two criteria: 1) the selected words should be interpretable and 2) they ought to suffice to reach nearly the same prediction (target vector) as the original input. In other words, a rationale must be short and sufficient. We will assume that a short selection is interpretable and focus on optimizing sufficiency under cardinality constraints.

We encapsulate the selection of words as a *rationale generator* which is another parameterized mapping $\mathbf{gen}(\mathbf{x})$ from input sequences to shorter sequences of words. Thus $\mathbf{gen}(\mathbf{x})$ must include only a few words and $\mathbf{enc}(\mathbf{gen}(\mathbf{x}))$ should result in nearly the same target vector as the original input passed through the encoder or $\mathbf{enc}(\mathbf{x})$. We can think of the generator as a tagging model where each word in the input receives a binary tag pertaining to whether it is selected to be included in the rationale. In our case, the generator is probabilistic and specifies a distribution over possible selections.

The rationale generation task is entirely unsupervised in the sense that we assume no explicit annotations about which words should be included in the rationale. Put another way, the rationale is introduced as a latent variable, a constraint that guides how to interpret the input sequence. The encoder and generator are trained jointly, in an end-to-end fashion so as to function well together.

4 Encoder and Generator

We use multi-aspect sentiment prediction as a guiding example to instantiate the two key components – the encoder and the generator. The framework itself generalizes to other tasks.

Encoder $\mathbf{enc}(\cdot)$: Given a training instance $(\mathbf{x}, \mathbf{y})$ where $\mathbf{x} = \{x_t\}_{t=1}^{l}$ is the input text sequence of length l and $\mathbf{y} \in [0, 1]^m$ is the target m-dimensional sentiment vector, the neural encoder predicts $\tilde{\mathbf{y}} = \mathbf{enc}(\mathbf{x})$. If trained on its own, the encoder would aim to minimize the discrepancy between the predicted sentiment vector $\tilde{\mathbf{y}}$ and the gold target vector $\mathbf{y}$. We will use the squared error (i.e. L_2 distance) as the sentiment loss function,

$$\mathcal{L}(\mathbf{x}, \mathbf{y}) = \|\tilde{\mathbf{y}} - \mathbf{y}\|_2^2 = \|\mathbf{enc}(\mathbf{x}) - \mathbf{y}\|_2^2$$

The encoder could be realized in many ways such as a recurrent neural network. For example, let $\mathbf{h}_t = f_e(\mathbf{x}_t, \mathbf{h}_{t-1})$ denote a parameterized recurrent unit mapping input word $\mathbf{x}_t$ and previous state $\mathbf{h}_{t-1}$ to next state $\mathbf{h}_t$. The target vector is then generated on the basis of the final state reached by the recurrent unit after processing all the words in the input sequence. Specifically,

$$\mathbf{h}_t = f_e(\mathbf{x}_t, \mathbf{h}_{t-1}), \;\; t = 1, \ldots, l$$
$$\tilde{\mathbf{y}} = \sigma_e(\mathbf{W}^e \mathbf{h}_l + \mathbf{b}^e)$$

Generator $\mathbf{gen}(\cdot)$: The rationale generator extracts a subset of text from the original input $\mathbf{x}$ to function as an interpretable summary. Thus the rationale for a given sequence $\mathbf{x}$ can be equivalently defined in terms of binary variables $\{\mathbf{z}_1, \cdots, \mathbf{z}_l\}$ where each $\mathbf{z}_t \in 0, 1$ indicates whether word $\mathbf{x}_t$ is selected or not. From here on, we will use $\mathbf{z}$ to specify the binary selections and thus $(\mathbf{z}, \mathbf{x})$ is the actual rationale generated (selections, input). We will use generator $\mathbf{gen}(\mathbf{x})$ as synonymous with a

probability distribution over binary selections, i.e., $\mathbf{z} \sim \mathbf{gen}(\mathbf{x}) \equiv p(\mathbf{z}|\mathbf{x})$ where the length of $\mathbf{z}$ varies with the input $\mathbf{x}$.

In a simple generator, the probability that the t^{th} word is selected can be assumed to be conditionally independent from other selections given the input $\mathbf{x}$. That is, the joint probability $p(\mathbf{z}|\mathbf{x})$ factors according to

$$p(\mathbf{z}|\mathbf{x}) = \prod_{t=1}^{l} p(\mathbf{z}_t|\mathbf{x}) \quad \text{(independent selection)}$$

The component distributions $p(\mathbf{z}_t|\mathbf{x})$ can be modeled using a shared bi-directional recurrent neural network. Specifically, let $\overrightarrow{f}()$ and $\overleftarrow{f}()$ be the forward and backward recurrent unit, respectively, then

$$\overrightarrow{\mathbf{h}_t} = \overrightarrow{f}(\mathbf{x}_t, \overrightarrow{\mathbf{h}_{t-1}})$$
$$\overleftarrow{\mathbf{h}_t} = \overleftarrow{f}(\mathbf{x}_t, \overleftarrow{\mathbf{h}_{t+1}})$$
$$p(\mathbf{z}_t|\mathbf{x}) = \sigma_z(\mathbf{W}^z[\overrightarrow{\mathbf{h}_t}; \overleftarrow{\mathbf{h}_t}] + \mathbf{b}^z)$$

Independent but context dependent selection of words is often sufficient. However, the model is unable to select phrases or refrain from selecting the same word again if already chosen. To this end, we also introduce a dependent selection of words,

$$p(\mathbf{z}|\mathbf{x}) = \prod_{t=1}^{l} p(\mathbf{z}_t|\mathbf{x}, \mathbf{z}_1 \cdots \mathbf{z}_{t-1})$$

which can be also expressed as a recurrent neural network. To this end, we introduce another hidden state $\mathbf{s}_t$ whose role is to couple the selections. For example,

$$p(\mathbf{z}_t|\mathbf{x}, \mathbf{z}_{1,t-1}) = \sigma_z(\mathbf{W}^z[\overrightarrow{\mathbf{h}_t}; \overleftarrow{\mathbf{h}_t}; \mathbf{s}_{t-1}] + \mathbf{b}^z)$$
$$\mathbf{s}_t = f_z([\overrightarrow{\mathbf{h}_t}; \overleftarrow{\mathbf{h}_t}; \mathbf{z}_t], \mathbf{s}_{t-1})$$

Joint objective: A rationale in our definition corresponds to the selected words, i.e., $\{\mathbf{x}_k|\mathbf{z}_k = 1\}$. We will use $(\mathbf{z}, \mathbf{x})$ as the shorthand for this rationale and, thus, $\mathbf{enc}(\mathbf{z}, \mathbf{x})$ refers to the target vector obtained by applying the encoder to the rationale as the input. Our goal here is to formalize how the rationale can be made short and meaningful yet function well in conjunction with the encoder. Our generator and encoder are learned jointly to interact well but they are treated as independent units for modularity.

The generator is guided in two ways during learning. First, the rationale that it produces must suffice as a replacement for the input text. In other words, the target vector (sentiment) arising from the rationale should be close to the gold sentiment. The corresponding loss function is given by

$$\mathcal{L}(\mathbf{z}, \mathbf{x}, \mathbf{y}) = \|\mathbf{enc}(\mathbf{z}, \mathbf{x}) - \mathbf{y}\|_2^2$$

Note that the loss function depends directly (parametrically) on the encoder but only indirectly on the generator via the sampled selection.

Second, we must guide the generator to realize short and coherent rationales. It should select only a few words and those selections should form phrases (consecutive words) rather than represent isolated, disconnected words. We therefore introduce an additional regularizer over the selections

$$\Omega(\mathbf{z}) = \lambda_1\|\mathbf{z}\| + \lambda_2 \sum_t |\mathbf{z}_t - \mathbf{z}_{t-1}|$$

where the first term penalizes the number of selections while the second one discourages transitions (encourages continuity of selections). Note that this regularizer also depends on the generator only indirectly via the selected rationale. This is because it is easier to assess the rationale once produced rather than directly guide how it is obtained.

Our final cost function is the combination of the two, $\text{cost}(\mathbf{z}, \mathbf{x}, \mathbf{y}) = \mathcal{L}(\mathbf{z}, \mathbf{x}, \mathbf{y}) + \Omega(\mathbf{z})$. Since the selections are not provided during training, we minimize the expected cost:

$$\min_{\theta_e, \theta_g} \sum_{(\mathbf{x},\mathbf{y}) \in D} \mathbb{E}_{\mathbf{z} \sim \mathbf{gen}(\mathbf{x})} \left[\text{cost}(\mathbf{z}, \mathbf{x}, \mathbf{y})\right]$$

where θ_e and θ_g denote the set of parameters of the encoder and generator, respectively, and D is the collection of training instances. Our joint objective encourages the generator to compress the input text into coherent summaries that work well with the associated encoder it is trained with.

Minimizing the expected cost is challenging since it involves summing over all the possible choices of rationales $\mathbf{z}$. This summation could potentially be made feasible with additional restrictive assumptions about the generator and encoder. However, we assume only that it is possible to efficiently sample from the generator.

Doubly stochastic gradient We now derive a sampled approximation to the gradient of the expected cost objective. This sampled approximation is obtained separately for each input text $\mathbf{x}$ so as to work well with an overall stochastic gradient method. Consider therefore a training pair $(\mathbf{x}, \mathbf{y})$. For the parameters of the generator θ_g,

$$\begin{aligned}
&\frac{\partial \mathbb{E}_{\mathbf{z}\sim\mathbf{gen}(\mathbf{x})}\left[\text{cost}(\mathbf{z},\mathbf{x},\mathbf{y})\right]}{\partial \theta_g} \\
&= \sum_{\mathbf{z}} \text{cost}(\mathbf{z},\mathbf{x},\mathbf{y}) \cdot \frac{\partial p(\mathbf{z}|\mathbf{x})}{\partial \theta_g} \\
&= \sum_{\mathbf{z}} \text{cost}(\mathbf{z},\mathbf{x},\mathbf{y}) \cdot \frac{\partial p(\mathbf{z}|\mathbf{x})}{\partial \theta_g} \cdot \frac{p(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})}
\end{aligned}$$

Using the fact $(\log f(\theta))' = f'(\theta)/f(\theta)$, we get

$$\begin{aligned}
&\sum_{\mathbf{z}} \text{cost}(\mathbf{z},\mathbf{x},\mathbf{y}) \cdot \frac{\partial p(\mathbf{z}|\mathbf{x})}{\partial \theta_g} \cdot \frac{p(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})} \\
&= \sum_{\mathbf{z}} \text{cost}(\mathbf{z},\mathbf{x},\mathbf{y}) \cdot \frac{\partial \log p(\mathbf{z}|\mathbf{x})}{\partial \theta_g} \cdot p(\mathbf{z}|\mathbf{x}) \\
&= \mathbb{E}_{z\sim\mathbf{gen}(\mathbf{x})}\left[\text{cost}(\mathbf{z},\mathbf{x},\mathbf{y}) \frac{\partial \log p(\mathbf{z}|\mathbf{x})}{\partial \theta_g}\right]
\end{aligned}$$

The last term is the expected gradient where the expectation is taken with respect to the generator distribution over rationales $\mathbf{z}$. Therefore, we can simply sample a few rationales $\mathbf{z}$ from the generator $\mathbf{gen}(\mathbf{x})$ and use the resulting average gradient in an overall stochastic gradient method. A sampled approximation to the gradient with respect to the encoder parameters θ_e can be derived similarly,

$$\begin{aligned}
&\frac{\partial \mathbb{E}_{\mathbf{z}\sim\mathbf{gen}(\mathbf{x})}\left[\text{cost}(\mathbf{z},\mathbf{x},\mathbf{y})\right]}{\partial \theta_e} \\
&= \sum_{\mathbf{z}} \frac{\partial \text{cost}(\mathbf{z},\mathbf{x},\mathbf{y})}{\partial \theta_e} \cdot p(\mathbf{z}|\mathbf{x}) \\
&= \mathbb{E}_{z\sim\mathbf{gen}(\mathbf{x})}\left[\frac{\partial \text{cost}(\mathbf{z},\mathbf{x},\mathbf{y})}{\partial \theta_e}\right]
\end{aligned}$$

Choice of recurrent unit We employ recurrent convolution (RCNN), a refinement of local-ngram based convolution. RCNN attempts to learn n-gram features that are not necessarily consecutive, and average features in a dynamic (recurrent) fashion. Specifically, for bigrams (filter width $n = 2$) RCNN computes $\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1})$ as follows

Number of reviews	1580k
Avg length of review	144.9
Avg correlation between aspects	63.5%
Max correlation between two aspects	79.1%
Number of annotated reviews	994

Table 1: Statistics of the beer review dataset.

$$\begin{aligned}
\lambda_t &= \sigma(\mathbf{W}^{\lambda}\mathbf{x}_t + \mathbf{U}^{\lambda}\mathbf{h}_{t-1} + \mathbf{b}^{\lambda}) \\
\mathbf{c}_t^{(1)} &= \lambda_t \odot \mathbf{c}_{t-1}^{(1)} + (1-\lambda_t) \odot (\mathbf{W}_1\mathbf{x}_t) \\
\mathbf{c}_t^{(2)} &= \lambda_t \odot \mathbf{c}_{t-1}^{(2)} + (1-\lambda_t) \odot (\mathbf{c}_{t-1}^{(1)} + \mathbf{W}_2\mathbf{x}_t) \\
\mathbf{h}_t &= \tanh(\mathbf{c}_t^{(2)} + \mathbf{b})
\end{aligned}$$

RCNN has been shown to work remarkably in classification and retrieval applications (Lei et al., 2015; Lei et al., 2016) compared to other alternatives such CNNs and LSTMs. We use it for all the recurrent units introduced in our model.

5 Experiments

We evaluate the proposed joint model on two NLP applications: (1) multi-aspect sentiment analysis on product reviews and (2) similar text retrieval on AskUbuntu question answering forum.

5.1 Multi-aspect Sentiment Analysis

Dataset We use the BeerAdvocate[2] review dataset used in prior work (McAuley et al., 2012).[3] This dataset contains 1.5 million reviews written by the website users. The reviews are naturally multi-aspect – each of them contains multiple sentences describing the *overall* impression or one particular aspect of a beer, including *appearance*, *smell* (aroma), *palate* and the *taste*. In addition to the written text, the reviewer provides the ratings (on a scale of 0 to 5 stars) for each aspect as well as an overall rating. The ratings can be fractional (e.g. 3.5 stars), so we normalize the scores to $[0, 1]$ and use them as the (only) supervision for regression.

McAuley et al. (2012) also provided sentence-level annotations on around 1,000 reviews. Each sentence is annotated with one (or multiple) aspect label, indicating what aspect this sentence covers.

[2] www.beeradvocate.com

[3] http://snap.stanford.edu/data/web-BeerAdvocate.html

Method	Appearance		Smell		Palate	
	% precision	% selected	% precision	% selected	% precision	% selected
SVM	38.3	13	21.6	7	24.9	7
Attention model	80.6	13	88.4	7	65.3	7
Generator (independent)	94.8	13	93.8	7	79.3	7
Generator (recurrent)	96.3	14	95.1	7	80.2	7

Table 2: Precision of selected rationales for the first three aspects. The precision is evaluated based on whether the selected words are in the sentences describing the target aspect, based on the sentence-level annotations. Best training epochs are selected based on the objective value on the development set (no sentence annotation is used).

	D	d	l	$\|\theta\|$	MSE
SVM	260k	-	-	2.5M	0.0154
SVM	1580k	-	-	7.3M	0.0100
LSTM	260k	200	2	644k	0.0094
RCNN	260k	200	2	323k	**0.0087**

Table 3: Comparing neural encoders with bigram SVM model. MSE is the mean squared error on the test set. D is the amount of data used for training and development. d stands for the hidden dimension, l denotes the depth of network and $|\theta|$ denotes the number of parameters (number of features for SVM).

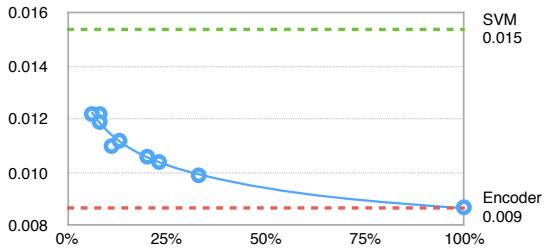


Figure 2: Mean squared error of all aspects on the test set (y-axis) when various percentages of text are extracted as rationales (x-axis). 220k training data is used.

We use this set as our test set to evaluate the precision of words in the extracted rationales.

Table 1 shows several statistics of the beer review dataset. The sentiment correlation between any pair of aspects (and the overall score) is quite high, getting 63.5% on average and a maximum of 79.1% (between the *taste* and *overall* score). If directly training the model on this set, the model can be confused due to such strong correlation. We therefore perform a preprocessing step, picking "less correlated" examples from the dataset.[4] This gives us a de-correlated subset for each aspect, each containing about 80k to 90k reviews. We use 10k as the development set. We focus on three aspects since the fourth aspect *taste* still gets $> 50\%$ correlation with the overall sentiment.

Sentiment Prediction Before training the joint model, it is worth assessing the neural encoder separately to check how accurately the neural network predicts the sentiment. To this end, we compare neural encoders with bigram SVM model, training medium and large SVM models using 260k and all 1580k reviews respectively. As shown in Table 3, the recurrent neural network models outperform the SVM model for sentiment prediction and also require less training data to achieve the performance. The LSTM and RCNN units obtain similar test error, getting 0.0094 and 0.0087 mean squared error respectively. The RCNN unit performs slightly better and uses less parameters. Based on the results, we choose the RCNN encoder network with 2 stacking layers and 200 hidden states.

To train the joint model, we also use RCNN unit with 200 states as the forward and backward recurrent unit for the generator $\mathbf{gen}()$. The dependent generator has one additional recurrent layer. For this layer we use 30 states so the dependent version still has a number of parameters comparable to the independent version. The two versions of the generator have 358k and 323k parameters respectively.

Figure 2 shows the performance of our joint dependent model when trained to predict the sentiment of all aspects. We vary the regularization λ_1 and λ_2 to show various runs that extract different amount of text as rationales. Our joint model gets performance close to the best encoder run (with full text) when few words are extracted.

[4] Specifically, for each aspect we train a simple linear regression model to predict the rating of this aspect given the ratings of the other four aspects. We then keep picking reviews with largest prediction error until the sentiment correlation in the selected subset increases dramatically.

a beer that is not sold in my neck of the woods , but managed to get while on a roadtrip . poured into an imperial pint glass with a generous head that sustained life throughout . nothing out of the ordinary here , but a good brew still . body was kind of heavy , but not thick . the hop smell was excellent and enticing . very drinkable

very dark beer . pours a nice finger and a half of creamy foam and stays throughout the beer . smells of coffee and roasted malt . has a major coffee-like taste with hints of chocolate . if you like black coffee , you will love this porter . creamy smooth mouthfeel and definitely gets smoother on the palate once it warms . it 's an ok porter but i feel there are much better one 's out there .

i really did not like this . it just seemed extremely watery . i dont ' think this had any carbonation whatsoever . maybe it was flat , who knows ? but even if i got a bad brew i do n't see how this would possibly be something i 'd get time and time again . i could taste the hops towards the middle , but the beer got pretty nasty towards the bottom . i would never drink this again , unless it was free . i 'm kind of upset i bought this .

a : poured a nice dark brown with a tan colored head about half an inch thick , nice red/garnet accents when held to the light . little clumps of lacing all around the glass , not too shabby . not terribly impressive though s : smells like a more guinness-y guinness really , there are some roasted malts there , signature guinness smells , less burnt though , a little bit of chocolate ... ... m : relatively thick , it is n't an export stout or imperial stout , but still is pretty hefty in the mouth , very smooth , not much carbonation . not too shabby d : not quite as drinkable as the draught , but still not too bad . i could easily see drinking a few of these .

Figure 3: Examples of extracted rationales indicating the sentiments of various aspects. The extracted texts for appearance, smell and palate are shown in red, blue and green color respectively. The last example is shortened for space.

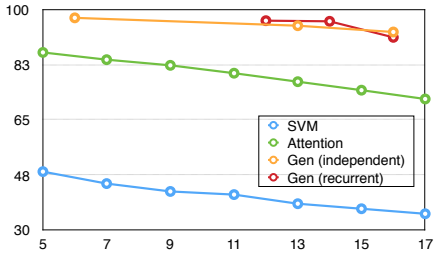


Figure 4: Precision (y-axis) when various percentages of text are extracted as rationales (x-axis) for the appearance aspect.

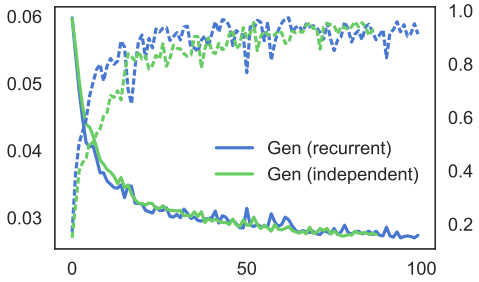


Figure 5: Learning curves of the optimized cost function on the development set and the precision of rationales on the test set. The smell (aroma) aspect is the target aspect.

Rationale Selection To evaluate the supporting rationales for each aspect, we train the joint encoder-generator model on each de-correlated subset. We set the cardinality regularization λ_1 between values $\{2e-4, 3e-4, 4e-4\}$ so the extracted rationale texts are neither too long nor too short. For simplicity, we set $\lambda_2 = 2\lambda_1$ to encourage local coherency of the extraction.

For comparison we use the bigram SVM model and implement an attention-based neural network model. The SVM model successively extracts unigram or bigram (from the test reviews) with the highest feature. The attention-based model learns a normalized attention vector of the input tokens (using similarly the forward and backward RNNs), then the model averages over the encoder states accordingly to the attention, and feed the averaged vector to the output layer. Similar to the SVM model, the attention-based model can selects words based on their attention weights.

Table 2 presents the precision of the extracted rationales calculated based on sentence-level aspect annotations. The λ_1 regularization hyper-parameter is tuned so the two versions of our model extract similar number of words as rationales. The SVM and attention-based model are constrained similarly for comparison. Figure 4 further shows the precision when different amounts of text are extracted. Again, for our model this corresponds to changing the λ_1 regularization. As shown in the table and the figure, our encoder-generator networks extract text pieces describing the target aspect with high precision, ranging from 80% to 96% across the three aspects appearance, smell and palate. The SVM baseline performs poorly, achieving around 30% accuracy. The attention-based model achieves reasonable but worse performance than the rationale generator, suggesting the potential of directly modeling rationales as explicit extraction.

Figure 5 shows the learning curves of our model for the smell aspect. In the early training epochs, both the independent and (recurrent) dependent selection models fail to produce good rationales, getting low precision as a result. After a few epochs of exploration however, the models start to achieve high accuracy. We observe that the dependent version learns more quickly in general, but both versions obtain close results in the end.

Finally we conduct a qualitative case study on the extracted rationales. Figure 3 presents several reviews, with highlighted rationales predicted by the model. Our rationale generator identifies key phrases or adjectives that indicate the sentiment of a particular aspect.

5.2 Similar Text Retrieval on QA Forum

Dataset For our second application, we use the real-world AskUbuntu[5] dataset used in recent work (dos Santos et al., 2015; Lei et al., 2016). This set contains a set of 167k unique questions (each consisting a question title and a body) and 16k user-identified similar question pairs. Following previous work, this data is used to train the neural encoder that learns the vector representation of the input question, optimizing the cosine distance (i.e. cosine similarity) between similar questions against random non-similar ones. We use the "one-versus-all" hinge loss (i.e. positive versus other negatives) for the encoder, similar to (Lei et al., 2016). During development and testing, the model is used to score 20 candidate questions given each query question, and a total of 400×20 query-candidate question pairs are annotated for evaluation[6].

Task/Evaluation Setup The question descriptions are often long and fraught with irrelevant details. In this set-up, a fraction of the original question text should be sufficient to represent its content, and be used for retrieving similar questions. Therefore, we will evaluate rationales based on the accuracy of the question retrieval task, assuming that better rationales achieve higher performance. To put this performance in context, we also report the accuracy when full body of a question is used, as well as titles alone. The latter constitutes an upper bound on the model performance as in this dataset titles provide short, informative summaries of the question content. We evaluate the rationales using the mean average precision (MAP) of retrieval.

	MAP (dev)	MAP (test)	%words
Full title	56.5	60.0	10.1
Full body	54.2	53.0	89.9
Independent	55.7	53.6	9.7
	56.3	52.6	19.7
Dependent	56.1	54.6	11.6
	56.5	55.6	32.8

Table 4: Comparison between rationale models (middle and bottom rows) and the baselines using full title or body (top row).

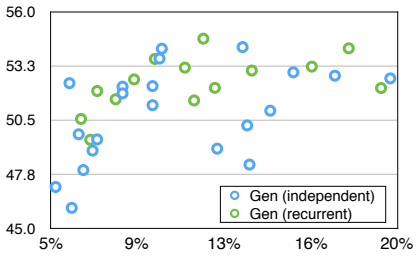


Figure 6: Retrieval MAP on the test set when various percentages of the texts are chosen as rationales. Data points correspond to models trained with different hyper-parameters.

Results Table 4 presents the results of our rationale model. We explore a range of hyper-parameter values[7]. We include two runs for each version. The first one achieves the highest MAP on the development set, The second run is selected to compare the models when they use roughly 10% of question text (7 words on average). We also show the results of different runs in Figure 6. The rationales achieve the MAP up to 56.5%, getting close to using the titles. The models also outperform the baseline of using the noisy question bodies, indicating the the models' capacity of extracting short but important fragments.

Figure 7 shows the rationales for several questions in the AskUbuntu domain, using the recurrent version with around 10% extraction. Interestingly, the model does not always select words from the question title. The reasons are that the question body can contain the same or even complementary information useful for retrieval. Indeed, some rationale fragments shown in the figure are error messages,

[5] askubuntu.com

[6] https://github.com/taolei87/askubuntu

[7] $\lambda_1 \in \{.008, .01, .012, .015\}$, $\lambda_2 = \{0, \lambda_1, 2\lambda_1\}$, dropout $\in \{0.1, 0.2\}$

what is the easiest way to install all the media codec available for ubuntu ? i am having issues with multiple applications prompting me to install codecs before they can play my files . how do i install media codecs ?

---

what should i do when i see <unk> report this <unk> ? an unresolvable problem occurred while initializing the package information . please report this bug against the 'update-manager ' package and include the following error message : e : encountered a section with no package : header e : problem with mergelist <unk> e : the package lists or status file could not be parsed or opened .

---

please any one give the solution for this whenever i try to convert the rpm file to deb file i always get this problem error : <unk> : not an rpm package (or package manifest) error executing `` lang=c rpm -qp -- queryformat % { name } <unk> ' '' : at <unk> line 489 thanks converting rpm file to debian fle

---

how do i mount a hibernated partition with windows 8 in ubuntu ? i ca n't mount my other partition with windows 8 , i have ubuntu 12.10 amd64 : error mounting /dev/sda1 at <unk> : command-line `mount -t `` ntfs '' -o `` uhelper=udisks2 , nodev , nosuid , uid=1000 , gid=1000 , dmask=0077 , fmask=0177 '' `` /dev/sda1 '' `` <unk> '' ' exited with non-zero exit status 14 : windows is hibernated , refused to mount . failed to mount '/dev/sda1 ' : operation not permitted the ntfs partition is hibernated . please resume and shutdown windows properly , or mount the volume read-only with the 'ro ' mount option

Figure 7: Examples of extracted rationales of questions in the AskUbuntu domain.

which are typically not in the titles but very useful to identify similar questions.

6 Discussion

We proposed a novel modular neural framework to automatically generate concise yet sufficient text fragments to justify predictions made by neural networks. We demonstrated that our encoder-generator framework, trained in an end-to-end manner, gives rise to quality rationales in the absence of any explicit rationale annotations. The approach could be modified or extended in various ways to other applications or types of data.

Choices of $\mathbf{enc}(\cdot)$ and $\mathbf{gen}(\cdot)$. The encoder and generator can be realized in numerous ways without changing the broader algorithm. For instance, we could use a convolutional network (Kim, 2014; Kalchbrenner et al., 2014), deep averaging network (Iyyer et al., 2015; Joulin et al., 2016) or a boosting classifier as the encoder. When rationales can be expected to conform to repeated stereotypical patterns in the text, a simpler encoder consistent with this bias can work better. We emphasize that, in this paper, rationales are flexible explanations that may vary substantially from instance to another. On the generator side, many additional constraints could be imposed to further guide acceptable rationales.

Dealing with Search Space. Our training method employs a REINFORCE-style algorithm (Williams, 1992) where the gradient with respect to the parameters is estimated by sampling possible rationales. Additional constraints on the generator output can be helpful in alleviating problems of exploring potentially a large space of possible rationales in terms of their interaction with the encoder. We could also apply variance reduction techniques to increase stability of stochastic training (cf. (Weaver and Tao, 2001; Mnih et al., 2014; Ba et al., 2015; Xu et al., 2015)).

7 Acknowledgments

We thank Prof. Julian McAuley for sharing the review dataset and annotations. We also thank MIT NLP group and the reviewers for their helpful comments. The work is supported by the Arabic Language Technologies (ALT) group at Qatar Computing Research Institute (QCRI) within the IYAS project. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors, and do not necessarily reflect the views of the funding organizations.

References

Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. 2015. Multiple object recognition with visual attention. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Kan Chen, Jiang Wang, Liang-Chieh Chen, Haoyuan Gao, Wei Xu, and Ram Nevatia. 2015. Abc-

cnn: An attention based convolutional neural network for visual question answering. *arXiv preprint arXiv:1511.05960*.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*.

Mark W Craven and Jude W Shavlik. 1996. Extracting tree-structured representations of trained networks. In *Advances in neural information processing systems (NIPS)*.

Cicero dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. 2015. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 694–699, Beijing, China, July. Association for Computational Linguistics.

Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015a. Retrofitting word vectors to semantic lexicons. In *Proceedings of NAACL*.

Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah A. Smith. 2015b. Sparse overcomplete word vector representations. In *Proceedings of ACL*.

Aurélie Herbelot and Eva Maria Vecchi. 2015. Building a shared world: mapping distributional to model-theoretic semantic spaces. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1684–1692.

Michiel Hermans and Benjamin Schrauwen. 2013. Training and analysing deep recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 190–198.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics*.

Andrej Karpathy, Justin Johnson, and Fei-Fei Li. 2015. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*.

B Kim, JA Shah, and F Doshi-Velez. 2015. Mind the gap: A generative approach to interpretable feature selection and extraction. In *Advances in Neural Information Processing Systems*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2015. Molding cnns for text: non-linear, non-consecutive convolutions. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi Jaakkola, Katerina Tymoshenko, Alessandro Moschitti, and Lluís Màrquez. 2016. Semi-supervised question retrieval with gated convolutions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*.

Benjamin Letham, Cynthia Rudin, Tyler H. McCormick, and David Madigan. 2015. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *Annals of Applied Statistics*, 9(3):1350–1371.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in nlp. In *Proceedings of NAACL*.

Iain J Marshall, Joël Kuiper, and Byron C Wallace. 2015. Robotreviewer: evaluation of a system for automatically assessing bias in clinical trials. *Journal of the American Medical Informatics Association*.

André F. T. Martins and Ramón Fernandez Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. *CoRR*, abs/1602.02068.

Julian McAuley, Jure Leskovec, and Dan Jurafsky. 2012. Learning attitudes and attributes from multi-aspect reviews. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 1020–1025. IEEE.

Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. 2014. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems (NIPS)*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?": Explaining the predictions of any classifier. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *International Conference on Learning Representations*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Sebastian Thrun. 1995. Extracting rules from artificial neural networks with distributed representations. In *Advances in neural information processing systems (NIPS)*.

Lex Weaver and Nigel Tao. 2001. The optimal reward baseline for gradient-based reinforcement learning. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*.

Huijuan Xu and Kate Saenko. 2015. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. *arXiv preprint arXiv:1511.05234*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. 2015. Stacked attention networks for image question answering. *arXiv preprint arXiv:1511.02274*.

Omar Zaidan, Jason Eisner, and Christine D. Piatko. 2007. Using "annotator rationales" to improve machine learning for text categorization. In *Proceedings of Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 260–267.

Ye Zhang, Iain James Marshall, and Byron C. Wallace. 2016. Rationale-augmented convolutional neural networks for text classification. *CoRR*, abs/1605.04469.

Semantic Understanding of Scenes through the ADE20K Dataset

Bolei Zhou[1], Hang Zhao[1], Xavier Puig[1], Sanja Fidler[2], Adela Barriuso[1], and Antonio Torralba[1]

[1]Massachusetts Institute of Technology, USA
[2]University of Toronto, Canada

Abstract

Scene parsing, or recognizing and segmenting objects and stuff in an image, is one of the key problems in computer vision. Despite the community's efforts in data collection, there are still few image datasets covering a wide range of scenes and object categories with dense and detailed annotations for scene parsing. In this paper, we introduce and analyze the ADE20K dataset, spanning diverse annotations of scenes, objects, parts of objects, and in some cases even parts of parts. A generic network design called Cascade Segmentation Module is then proposed to enable the segmentation networks to parse a scene into stuff, objects, and object parts in a cascade. We evaluate the proposed module integrated within two existing semantic segmentation networks, yielding significant improvements for scene parsing. We further show that the scene parsing networks trained on ADE20K can be applied to a wide variety of scenes and objects[1].

1. Introduction

Semantic understanding of visual scenes is one of the holy grails of computer vision. The emergence of large-scale image datasets like ImageNet [20], COCO [14] and Places [26], along with the rapid development of the deep convolutional neural network (ConvNet) approaches, have brought great advancements to visual scene understanding. Nowadays, given a visual scene of a living room, a robot equipped with a trained ConvNet can accurately predict the scene category. However, to freely navigate in the scene and manipulate the objects inside, the robot has far more information to digest. It needs to recognize and localize not only the objects like sofa, table, and TV, but also their parts, e.g., a seat of a chair or a handle of a cup, to allow proper interaction, as well as to segment the stuff like floor, wall and ceiling for spatial navigation.

Scene parsing, or recognizing and segmenting objects and stuff in an image, remains one of the key problems in scene understanding. Going beyond the image-level recognition, scene parsing requires a much denser annotation of scenes with a large set of objects. However, the current datasets have limited number of objects (e.g., COCO [14], Pascal [9]) and in many cases those objects are not the most common objects one encounters in the world (like frisbees or baseball bats), or the datasets only cover a limited set of scenes (e.g., Cityscapes [6]). Some notable exceptions are Pascal-Context [17] and the SUN database [24]. However, Pascal-Context still contains scenes primarily focused on 20 object classes, while SUN has noisy labels at the object level.

Our goal is to collect a dataset that has densely annotated images (every pixel has a semantic label) with a large and an unrestricted open vocabulary. The images in our dataset are manually segmented in great detail, covering a diverse set of scenes, object and object part categories. The challenge for collecting such annotations is finding reliable annotators, as well as the fact that labeling is difficult if the class list is not defined in advance. On the other hand, open vocabulary naming also suffers from naming inconsistencies across different annotators. In contrast, our dataset was annotated by a single expert annotator, providing extremely detailed and exhaustive image annotations. On average, our annotator labeled 29 annotation segments per image, compared to the 16 segments per image labeled by external annotators (like workers from Amazon Mechanical Turk). Furthermore, the data consistency and quality are much higher than that of external annotators. Fig. 1 shows examples from our dataset.

The paper is organized as follows. Firstly we describe the ADE20K dataset, the collection process and statistics. We then introduce a generic network design called Cascade Segmentation Module, which enables neural networks to segment stuff, objects, and object parts in cascade. Several semantic segmentation networks are evaluated on the scene

[1]Dataset is available at `http://groups.csail.mit.edu/vision/datasets/ADE20K/`.

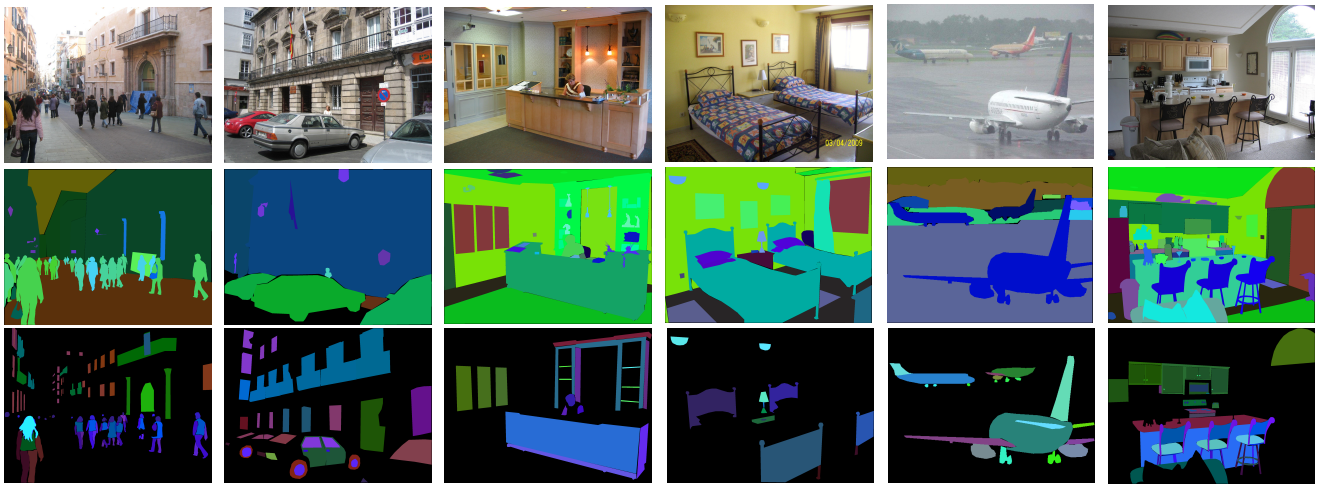


Figure 1. Images in ADE20K dataset are densely annotated in detail with objects and parts. The first row shows the sample images, the second row shows the annotation of objects, and the third row shows the annotation of object parts.

parsing benchmark of ADE20K as baselines. The proposed Cascade Segmentation Module is shown to improve over those baselines. We further apply the scene parsing networks to the tasks of hierarchical semantic segmentation and automatic scene content removal.

1.1. Related work

Many datasets have been collected for the purpose of semantic understanding of scenes. We review the datasets according to the level of details of their annotations, then briefly go through the previous work of semantic segmentation networks.

Object classification/detection datasets. Most of the large-scale datasets typically only contain labels at the image level or provide bounding boxes. Examples include Imagenet [20], Pascal [9], and KITTI [10]. Imagenet has the largest set of classes, but contains relatively simple scenes. Pascal and KITTI are more challenging and have more objects per image, however, their classes as well as scenes are more constrained.

Semantic segmentation datasets. Existing datasets with pixel-level labels typically provide annotations only for a subset of foreground objects (20 in PASCAL VOC [9] and 91 in Microsoft COCO [14]). Collecting dense annotations where all pixels are labeled is much more challenging. Such efforts include Pascal-Context [17], NYU Depth V2 [18], SUN database [24], SUN RGB-D dataset [22], CityScapes dataset [6], and OpenSurfaces [2, 3].

Datasets with objects, parts and attributes. Recently, two datasets were released that go beyond the typical labeling setup by also providing pixel-level annotation for the object parts, i.e. Pascal-Part dataset [5], or material classes, i.e. OpenSurfaces [2, 3]. We advance this effort by collecting very high-resolution imagery of a much wider selection of scenes, containing a large set of object classes per image. We annotated both stuff and object classes, for which we additionally annotated their parts, and parts of these parts. We believe that our dataset, ADE20K, is one of the most comprehensive datasets of its kind. We provide a comparison between datasets in Sec. 2.5.

Semantic segmentation networks. With the success of convolutional neural networks (CNN) for image classification [13], there is growing interest for semantic pixel-wise labeling using CNNs with dense output, such as the fully CNN [15], deconvolutional neural networks [19], encoder-decoder SegNet [1], multi-task network cascades [8], and DilatedNet [4, 25]. They are benchmarked on Pascal dataset with impressive performance on segmenting the 20 object classes. Some of them [15, 1] are evaluated on Pascal-Context [17] or SUN RGB-D dataset [22] to show the capability to segment more object classes in scenes. Joint stuff and object segmentation is explored in [7] which uses pre-computed superpixels and feature masking to represent stuff. Cascade of instance segmentation and categorization has been explored in [8]. In this paper we introduce a generic network module to segment stuff, objects, and object parts jointly in a cascade, which could be integrated in existing networks.

2. ADE20K: Fully Annotated Image Dataset

In this section, we describe our ADE20K dataset and analyze it through a variety of informative statistics.

2.1. Dataset summary

There are 20,210 images in the training set, 2,000 images in the validation set, and 3,000 images in the testing set. All the images are exhaustively annotated with objects. Many objects are also annotated with their parts. For each object

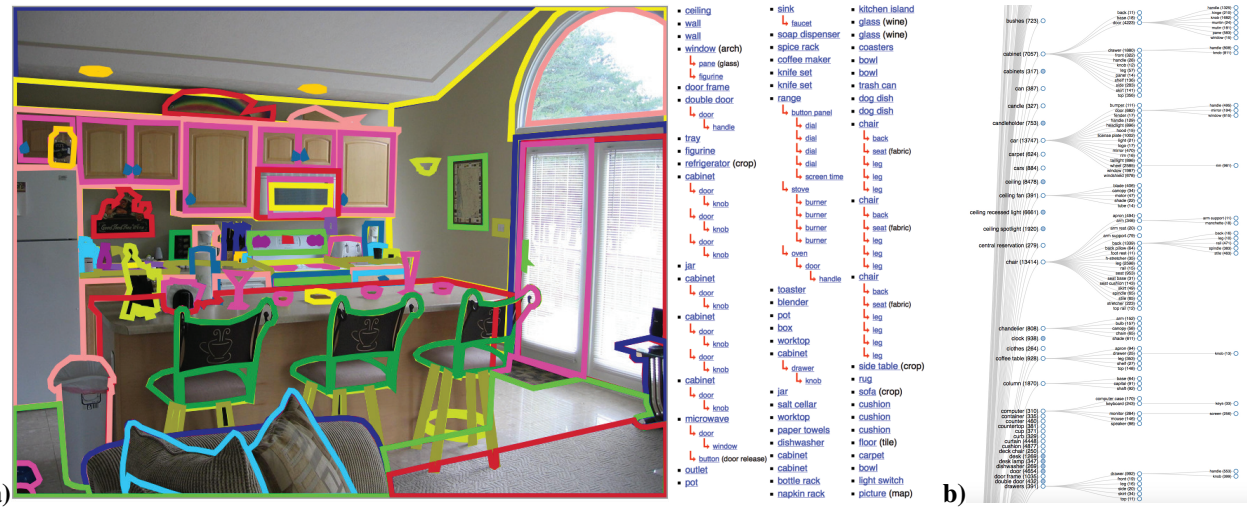


Figure 2. a) Annotation interface, the list of the objects and their associated parts in the image. b) Section of the relation tree of objects and parts for the dataset (see the dataset webpage for the full relation tree)

there is additional information about whether it is occluded or cropped, and other attributes. The images in the validation set are exhaustively annotated with parts, while the part annotations are not exhaustive over the images in the training set. The annotations in the dataset are still growing. Sample images and annotations from the ADE20K dataset are shown in Fig. 1.

2.2. Image annotation

For our dataset, we are interested in having a diverse set of scenes with dense annotations of all the objects present. Images come from the LabelMe [21], SUN datasets [24], and Places [26] and were selected to cover the 900 scene categories defined in the SUN database. Images were annotated by a single expert worker using the LabelMe interface [21]. Fig. 2.a shows a snapshot of the annotation interface and one fully segmented image. The worker provided three types of annotations: object segments with names, object parts, and attributes. All object instances are segmented independently so that the dataset could be used to train and evaluate detection or segmentation algorithms. Datasets such as COCO [14], Pascal [9] or Cityscape [6] start by defining a set of object categories of interest. However, when labeling all the objects in a scene, working with a predefined list of objects is not possible as new categories appear frequently (see fig. 5.d). Here, the annotator created a dictionary of visual concepts where new classes were added constantly to ensure consistency in object naming.

Object parts are associated with object instances. Note that parts can have parts too, and we label these associations as well. For example, the 'rim' is a part of a 'wheel', which in turn is part of a 'car'. A 'knob' is a part of a 'door' that can be part of a 'cabinet'. This part hierarchy in Fig. 2.b has a depth of 3.

2.3. Annotation consistency

Defining a labeling protocol is relatively easy when the labeling task is restricted to a fixed list of object classes, however it becomes challenging when the class list is open-ended. As the goal is to label all the objects within each image, the list of classes grows unbounded. Many object classes appear only a few times across the entire collection of images. However, those rare object classes cannot be ignored as they might be important elements for the interpretation of the scene. Labeling in these conditions becomes difficult because we need to keep a growing list of all the object classes in order to have a consistent naming across the entire dataset. Despite the annotator's best effort, the process is not free of noise.

To analyze the annotation consistency we took a subset of 61 randomly chosen images from the validation set, then asked our annotator to annotate them again (there is a time difference of six months). One expects that there are some differences between the two annotations. A few examples are shown in Fig 3. On average, 82.4% of the pixels got the same label. The remaining 17.6% of pixels had some errors for which we grouped into three error types as follows:

- **Segmentation quality**: Variations in the quality of segmentation and outlining of the object boundary. One typical source of error arises when segmenting complex objects such as buildings and trees, which can be segmented with different degrees of precision. 5.7% of the pixels had this type of error.

- **Object naming**: Differences in object naming (due to ambiguity or similarity between concepts, for instance, calling a big car a 'car' in one segmentation and a 'truck' in the another one, or a 'palm tree' a 'tree'.

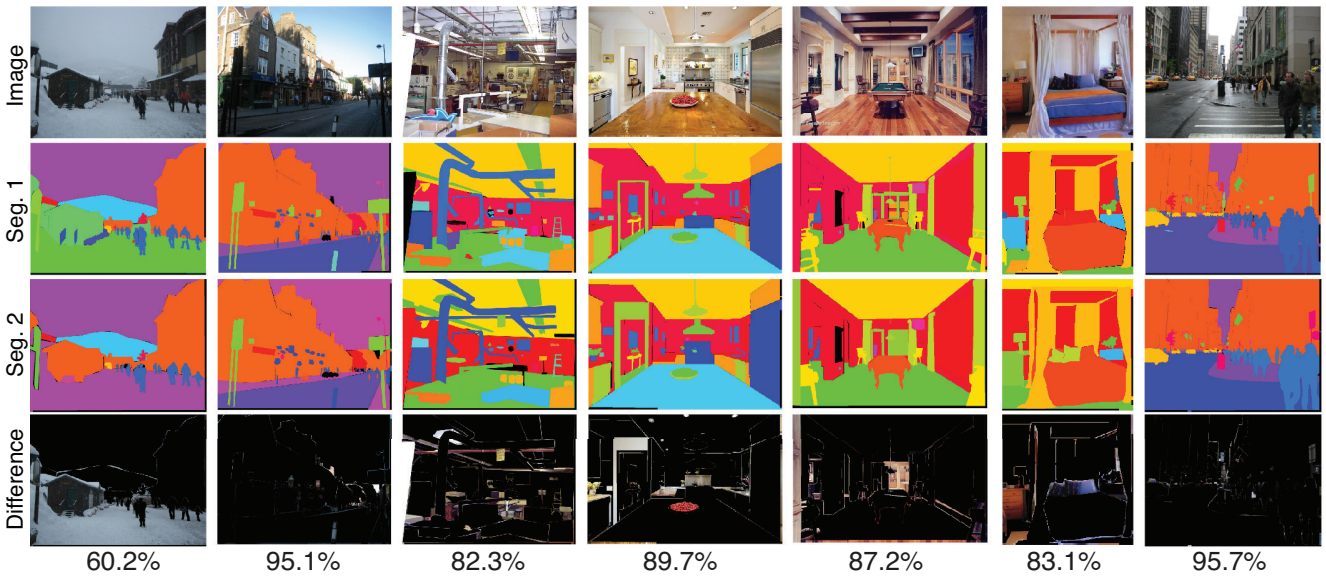


Figure 3. Analysis of annotation consistency. Each column shows an image and two segmentations done by the same annotator at different times. Bottom row shows the pixel discrepancy when the two segmentations are subtracted, while the number at the bottom shows the percentage of pixels with the same label. On average across all re-annotated images, 82.4% of pixels got the same label. In the example in the first column the percentage of pixels with the same label is relatively low because the annotator labeled the same region as 'snow' and 'ground' during the two rounds of annotation. In the third column, there were many objects in the scene and the annotator missed some between the two segmentations.

6.0% of the pixels had naming issues. These errors can be reduced by defining a very precise terminology, but this becomes much harder with a large growing vocabulary.

- **Segmentation quantity**: Missing objects in one of the two segmentations. There is a very large number of objects in each image and some images might be annotated more thoroughly than others. For example, in the third column of Fig 3 the annotator missed some small objects in different annotations. 5.9% of the pixels are due to missing labels. A similar issue existed in segmentation datasets such as the Berkeley Image segmentation dataset [16].

The median error values for the three error types are: 4.8%, 0.3% and 2.6% showing that the mean value is dominated by a few images, and that the most common type of error is segmentation quality.

To further compare the annotation done by our single expert annotator and the AMT-like annotators, 20 images from the validation set are annotated by two invited external annotators, both with prior experience in image labeling. The first external annotator had 58.5% of inconsistent pixels compared to the segmentation provided by our annotator, and the second external annotator had 75% of the inconsistent pixels. Many of these inconsistencies are due to the poor quality of the segmentations provided by external annotators (as it has been observed with AMT which requires multiple verification steps for quality control [14]). For the best external annotator (the first one), 7.9% of pixels have inconsistent segmentations (just slightly worse than our annotator), 14.9% have inconsistent object naming and 35.8% of the pixels correspond to missing objects, which is due to the much smaller number of objects annotated by the external annotator in comparison with the ones annotated by our expert annotator. The external annotators labeled on average 16 segments per image while our annotator provided 29 segments per image.

2.4. Dataset statistics

Fig. 4.a shows the distribution of ranked object frequencies. The distribution is similar to a Zipf's law and is typically found when objects are exhaustively annotated in images [23, 24]. They differ from the ones from datasets such as COCO or ImageNet where the distribution is more uniform resulting from manual balancing.

Fig. 4.b shows the distributions of annotated parts grouped by the objects they belong to and sorted by frequency within each object class. Most object classes also have a non-uniform distribution of part counts. Fig. 4.c and Fig. 4.d show how objects are shared across scenes and how parts are shared by objects. Fig. 4.e shows the variability in the appearances of the part 'door'.

The mode of the object segmentations is shown in Fig. 5.a and contains the four objects (from top to bottom): 'sky', 'wall', 'building' and 'floor'. When using simply the

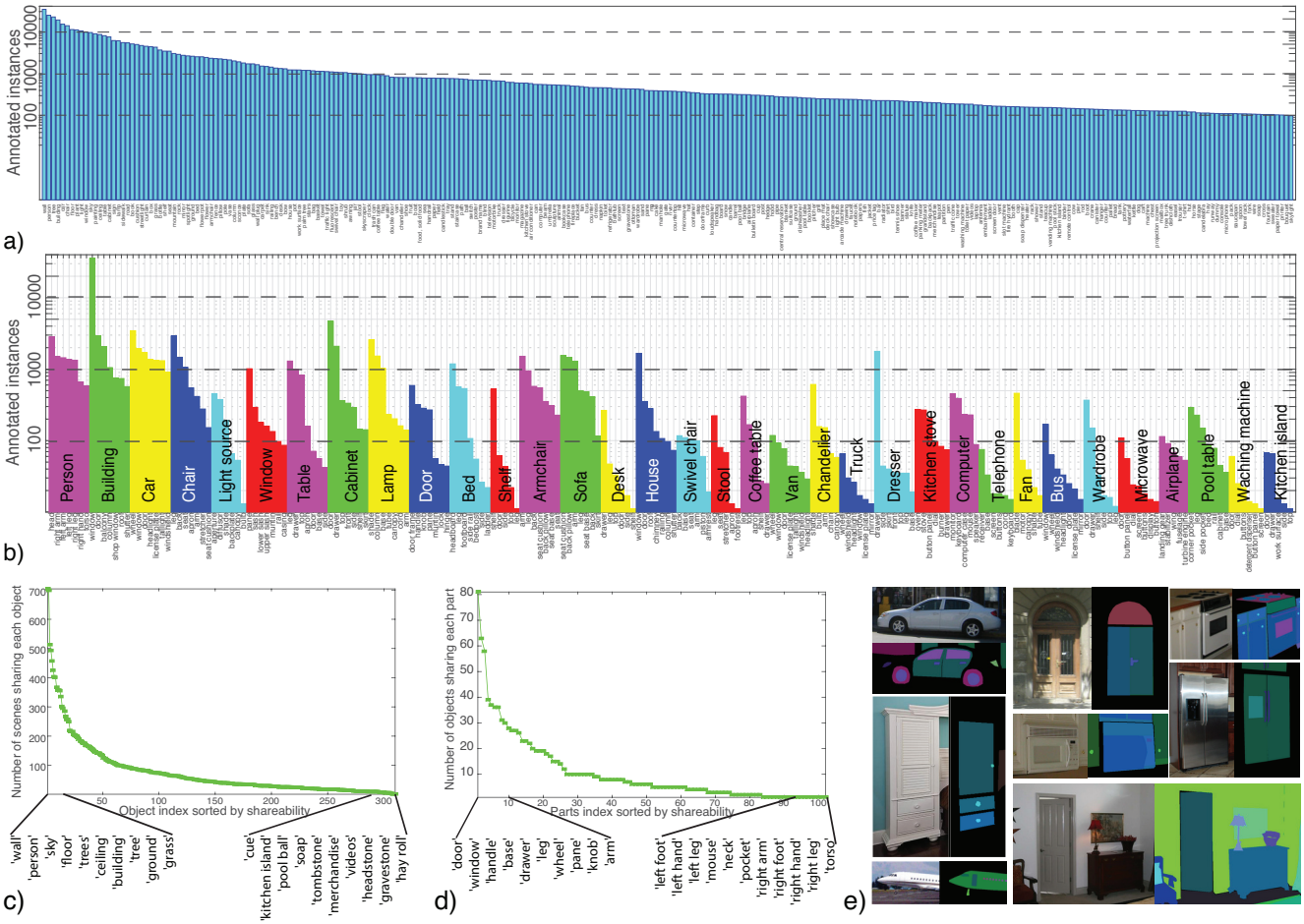


Figure 4. a) Object classes sorted by frequency. Only the top 270 classes with more than 100 annotated instances are shown. 68 classes have more than a 1000 segmented instances. b) Frequency of parts grouped by objects. There are more than 200 object classes with annotated parts. Only objects with 5 or more parts are shown in this plot (we show at most 7 parts for each object class). c) Objects ranked by the number of scenes they are part of. d) Object parts ranked by the number of objects they are part of. e) Examples of objects with doors. The bottom-right image is an example where the door does not behave as a part.

mode to segment the images, it gets, on average, 20.9% of the pixels of each image right. Fig. 5.b shows the distribution of images according to the number of distinct classes and instances. On average there are 19.5 instances and 10.5 object classes per image, larger than other existing datasets (see Table 1). Fig. 5.c shows the distribution of parts.

As the list of object classes is not predefined, there are new classes appearing over time of annotation. Fig. 5.d shows the number of object (and part) classes as the number of annotated instances increases. Fig. 5.e shows the probability that instance $n+1$ is a new class after labeling n instances. The more segments we have, the smaller the probability that we will see a new class. At the current state of the dataset, we get one new object class every 300 segmented instances.

2.5. Comparison with other datasets

We compare ADE20K with existing datasets in Tab. 1. Compared to the largest annotated datasets, COCO [14] and Imagenet [20], our dataset comprises of much more diverse scenes, where the average number of object classes per image is 3 and 6 times larger, respectively. With respect to SUN [24], ADE20K is roughly 35% larger in terms of images and object instances. However, the annotations in our dataset are much richer since they also include segmentation at the part level. Such annotation is only available for the Pascal-Context/Part dataset [17, 5] which contains 40 distinct part classes across 20 object classes. Note that we merged some of their part classes to be consistent with our labeling (e.g., we mark both *left leg* and *right leg* as the same semantic part *leg*). Since our dataset contains part annotations for a much wider set of object classes, the number of

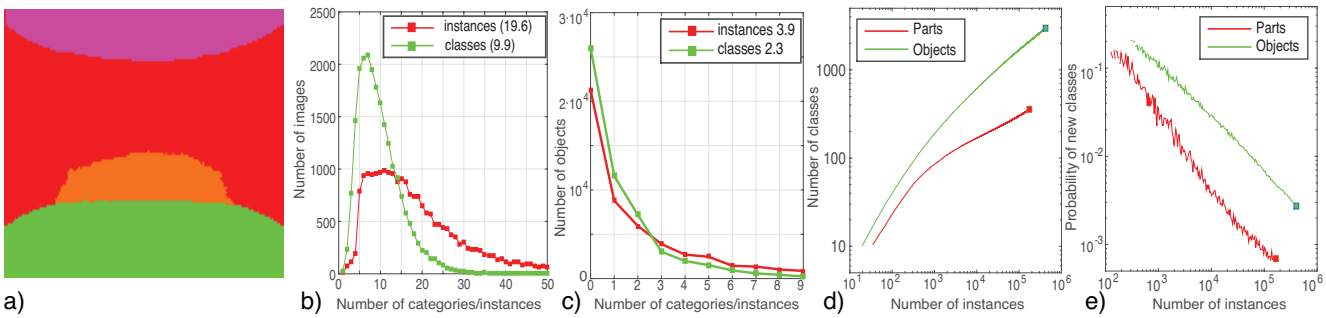


Figure 5. a) Mode of the object segmentations contains 'sky', 'wall', 'building' and 'floor'. b) Histogram of the number of segmented object instances and classes per image. c) Histogram of the number of segmented part instances and classes per object. d) Number of classes as a function of segmented instances (objects and parts). The squares represent the current state of the dataset. e) Probability of seeing a new object (or part) class as a function of the number of instances.

part classes is almost 9 times larger in our dataset.

An interesting fact is that any image in ADE20K contains at least 5 objects, and the maximum number of object instances per image reaches 273, and 419 instances, when counting parts as well. This shows the high annotation complexity of our dataset.

3. Cascade Segmentation Module

While the frequency of objects appearing in scenes follows a long-tail distribution, the pixel ratios of objects also follow such a distribution. For example, the stuff classes like 'wall', 'building', 'floor', and 'sky' occupy more than 40% of all the annotated pixels, while the discrete objects, such as 'vase' and 'microwave' at the tail of the distribution (see Fig. 4b), occupy only 0.03% of the annotated pixels. Because of the long-tail distribution, a semantic segmentation network can be easily dominated by the most frequent stuff classes. On the other hand, there are spatial layout relations among stuff and objects, and the objects and the object parts, which are ignored by the design of the previous semantic segmentation networks. For example, a drawing on a wall is a part of the wall (the drawing occludes the wall), and the wheels on a car are also parts of the car.

To handle the long-tail distribution of objects in scenes and the spatial layout relations of scenes, objects, and object parts, we propose a network design called Cascade Segmentation Module. This module is a generic network design which can potentially be integrated in any previous semantic segmentation networks. We first categorize semantic classes of the scenes into three macro classes: *stuff* (sky, road, building, etc), foreground *objects* (car, tree, sofa, etc), and object *parts* (car wheels and door, people head and torso, etc). Note that in some scenarios there are some object classes like 'building' or 'door' could belong to either of two macro classes, here we assign the object classes to their most likely macro class.

In the network for scene parsing, different streams of high-level layers are used to represent each macro class and recognize the assigned classes. The segmentation results from each stream are then fused to generate the segmentation. The proposed module is illustrated in Fig. 6.

More specifically, the stuff stream is trained to classify all the stuff classes plus one foreground object class (which includes all the non-stuff classes). After training, the stuff stream generates stuff segmentation and a dense objectness map indicating the probability that a pixel belongs to the foreground object class. The object stream is trained to classify the discrete objects. All the non-discrete objects are ignored in the training loss function of the object stream. After training, the object stream further segments each discrete object on the predicted objectness map from the stuff stream. The result is merged with the stuff segmentation to generate the scene segmentation. For those discrete objects annotated with parts, the part stream can be jointly trained to segment object parts. Thus the part stream further segments parts on each object score map predicted from the object stream.

The network with the two streams (stuff+objects) or three streams (stuff+objects+parts) could be trained end-to-end. The streams share the weights of the lower layers. Each stream has a training loss at the end. For the stuff stream we use the per-pixel cross-entropy loss, where the output classes are all the stuff classes plus the foreground class (all the discrete object classes are included in it). We use the per-pixel cross-entropy loss for the object stream, where the output classes are all the discrete object classes. The objectness map is given as a ground-truth binary mask that indicates whether a pixel belongs to any of the stuff classes or the foreground object class. This mask is used to exclude the penalty for pixels which belong to any of the stuff classes in the training loss for the object stream. Similarly, we use cross-entropy loss for the part stream. All part classes are trained together, while non-part pixels are ignored in training. In testing, parts are segmented on their

Table 1. Comparison with existing datasets with semantic segmentation.

	Images	Obj. inst.	Obj. classes	Part inst.	Part classes	Obj. classes per image
COCO	123,287	886,284	91	0	0	3.5
ImageNet*	476,688	534,309	200	0	0	1.7
NYU Depth V2	1,449	34,064	894	0	0	14.1
Cityscapes	25,000	65,385	30	0	0	12.2
SUN	16,873	313,884	4,479	0	0	9.8
OpenSurfaces	22,214	71,460	160	0	0	N/A
PascalContext	10,103	∼104,398**	540	181,770	40	5.1
ADE20K	22,210	434,826	2,693	175,961	476	9.9

* has only bounding boxes (no pixel-level segmentation). Sparse annotations.
** PascalContext dataset does not have instance segmentation. In order to estimate the number of instances, we find connected components (having at least 150pixels) for each class label.

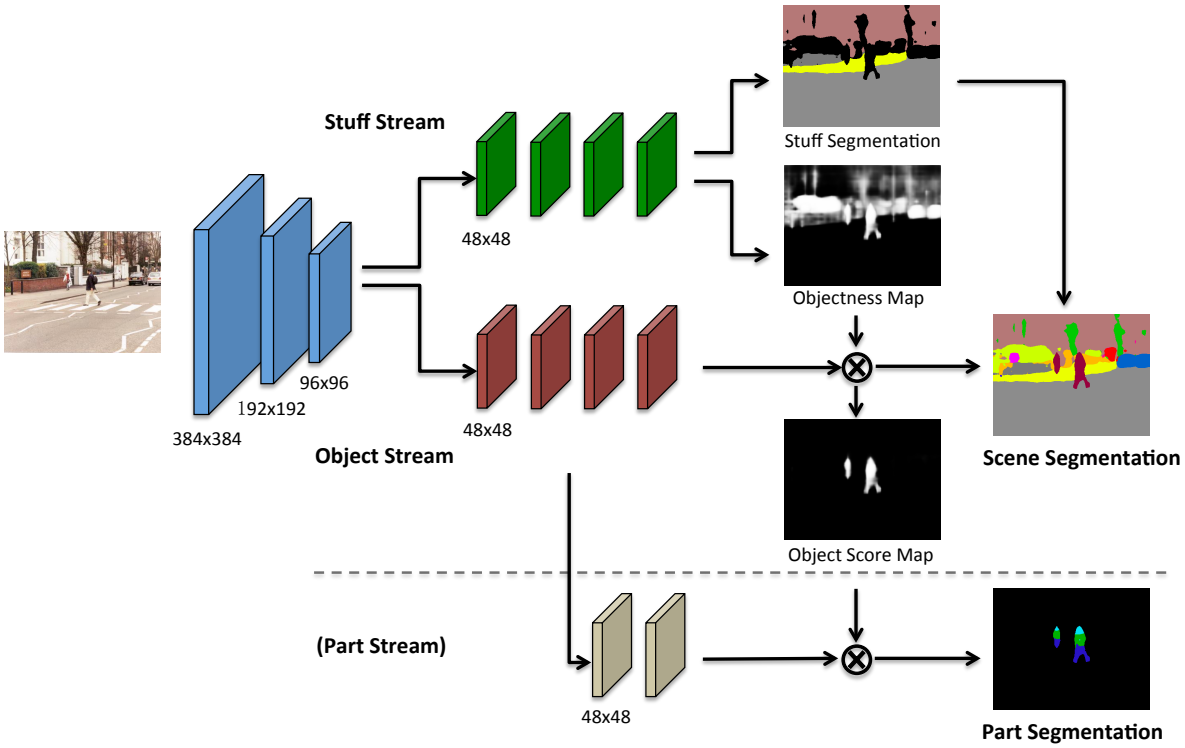


Figure 6. The framework of Cascade Segmentation Module for scene parsing. *Stuff* stream generates the stuff segmentation and objectness map from the shared feature activation. The *object* stream then generates object segmentation by integrating the objectness map from the stuff stream. Finally the full scene segmentation is generated by merging the object segmentation and the stuff segmentation. Similarly, *part* stream takes object score map from object stream to further generate object-part segmentation. Since not all objects have part annotation, the part stream is optional. Feature sizes are based on the Cascade-dilatedNet, the Cascade-SegNet has different but similar structures.

associated object score map given by the object stream. The training losses for the two streams and for the three streams are $L = L_{stuff} + L_{object}$ and $L = L_{stuff} + L_{object} + L_{part}$ respectively.

The configurations of each layer are based on the baseline network being used. We integrate the proposed module on two baseline networks Segnet [1] and DilatedNet [4, 25]. In the following experiments, we evaluate that the proposed module brings great improvements for scene parsing.

4. Experiments

To train the networks for scene parsing, we select the top 150 objects ranked by their total pixel ratios[2] from the ADE20K dataset and build a scene parsing benchmark of ADE20K, termed as SceneParse150. Among the 150 objects, there are 35 stuff classes (i.e., wall, sky, road) and 115 discrete objects (i.e., car, person, table). The annotated pix-

[2]As the original images in the ADE20K dataset have various sizes, for simplicity we rescale those large-sized images to make their minimum heights or widths as 512 in the SceneParse150 benchmark.

els of the 150 objects occupy 92.75% of all the pixels in the dataset, where the stuff classes occupy 60.92%, and discrete objects occupy 31.83%.

We map the wordnet synsets with each one of the object names, then build up a wordnet tree through the hypernym relations of the 150 objects shown in Fig. 7. We can see that these objects form several semantic clusters in the tree, such as the *furniture* synset node containing cabinet, desk, pool table, and bench, the *conveyance* node containing car, truck, boat, and bus, as well as the *living thing* node containing shrub, grass, flower, and person. Thus, the structured object annotation given in the dataset bridge the image annotation to a wider knowledge base.

4.1. Scene parsing

As for baselines of scene parsing on SceneParse150 benchmark, we train three semantic segmentation networks: SegNet [1], FCN-8s [15], and DilatedNet [4, 25]. SegNet has encoder and decoder architecture for image segmentation; FCN upsamples the activations of multiple layers in the CNN for pixelwise segmentation; DilatedNet drops *pool4* and *pool5* from fully convolutional VGG-16 network, and replaces the following convolutions with dilated convolutions (or atrous convolutions).

We integrate the proposed cascade segmentation module on the two baseline networks: SegNet and DilatedNet. We did not integrate it with FCN because the original FCN requires a large amount of GPU memory and has skip connections across layers. For the Cascade-SegNet, two streams share a single encoder, from `conv1_1` to `conv5_3`, while each stream has its own decoder, from `deconv5_3` to `loss`. For the Cascade-DilatedNet, the two streams split after `pool3`, and keep spatial dimensions of their feature maps afterwards. For a fair comparison and benchmark purposes, the cascade networks only have stuff stream and object stream. We train these network models using the Caffe library [12] on NVIDIA Titan X GPUs.

Results are reported in four metrics commonly used for semantic segmentation [15]:

- **Pixel accuracy** indicates the proportion of correctly classified pixels;
- **Mean accuracy** indicates the proportion of correctly classified pixels averaged over all the classes.
- **Mean IoU** indicates the intersection-over-union between the predicted and ground-truth pixels, averaged over all the classes.
- **Weighted IoU** indicates the IoU weighted by the total pixel ratio of each class.

Since some classes like 'wall' and 'floor' occupy far more pixels of the images, pixel accuracy is biased to reflect the accuracy over those few large classes. Instead, mean IoU reflects how accurately the model classifies each discrete class in the benchmark. The scene parsing data and the development toolbox are released in the Scene Parsing Challenge hosted at ILSVRC'16[3]. We take the average of the pixel accuracy and mean IoU as the evaluation criteria in the challenge.

Table 2. Performance on the validation set of SceneParse150.

Networks	Pixel Acc.	Mean Acc.	Mean IoU	Weighted IoU
FCN-8s	71.32%	40.32%	0.2939	0.5733
SegNet	71.00%	31.14%	0.2164	0.5384
DilatedNet	73.55%	44.59%	0.3231	0.6014
Cascade-SegNet	71.83%	37.90%	0.2751	0.5805
Cascade-DilatedNet	**74.52%**	**45.38%**	**0.3490**	**0.6108**

Table 3. Performance of stuff and discrete object segmentation.

	35 stuff		115 discrete objects	
Networks	Mean Acc.	Mean IoU	Mean Acc.	Mean IoU
FCN-8s	46.74%	0.3344	38.36%	0.2816
SegNet	43.17%	0.3051	27.48%	0.1894
DilatedNet	49.03%	0.3729	43.24%	0.3080
Cascade-SegNet	40.46%	0.3245	37.12%	0.2600
Cascade-DilatedNet	**49.80%**	**0.3779**	**44.04%**	**0.3401**

The segmentation results of the baselines and the cascade networks are listed in Table 2. Among the baselines, the DilatedNet achieves the best performance on the SceneParse150. The cascade networks, Cascade-SegNet and Cascade-DilatedNet both outperform the original baselines. In terms of mean IoU, the improvement brought by the proposed cascade segmentation module for SegNet is 6%, and for DilatedNet is 2.5%. We further decompose the performance of networks on 35 stuff and 115 discrete object classes respectively, in Table 3. We observe that the two cascade networks perform much better on the 115 discrete objects compared to the baselines. This validates that the design of cascade module helps improve scene parsing for the discrete objects as they have less training data but more visual complexity compared to those stuff classes.

Segmentation examples from the validation set are shown in Fig. 8. Compared to the baseline SegNet and DilatedNet, the segmentation results from the Cascade-SegNet and Cascade-DilatedNet are more detailed. Furthermore, the objectness maps from the stuff stream highlight the possible discrete objects in the scenes.

4.2. Part segmentation

For part segmentation, we select the eight object classes frequently annotated with parts: 'person', 'building', 'car', 'chair','table', 'sofa', 'bed', 'lamp'. After we filter out the part classes of those objects with instance number smaller than 300, there are 36 part classes included in the training and testing. We train the part stream on the Cascade-DilatedNet used in the scene parsing.

[3] `http://sceneparsing.csail.mit.edu`

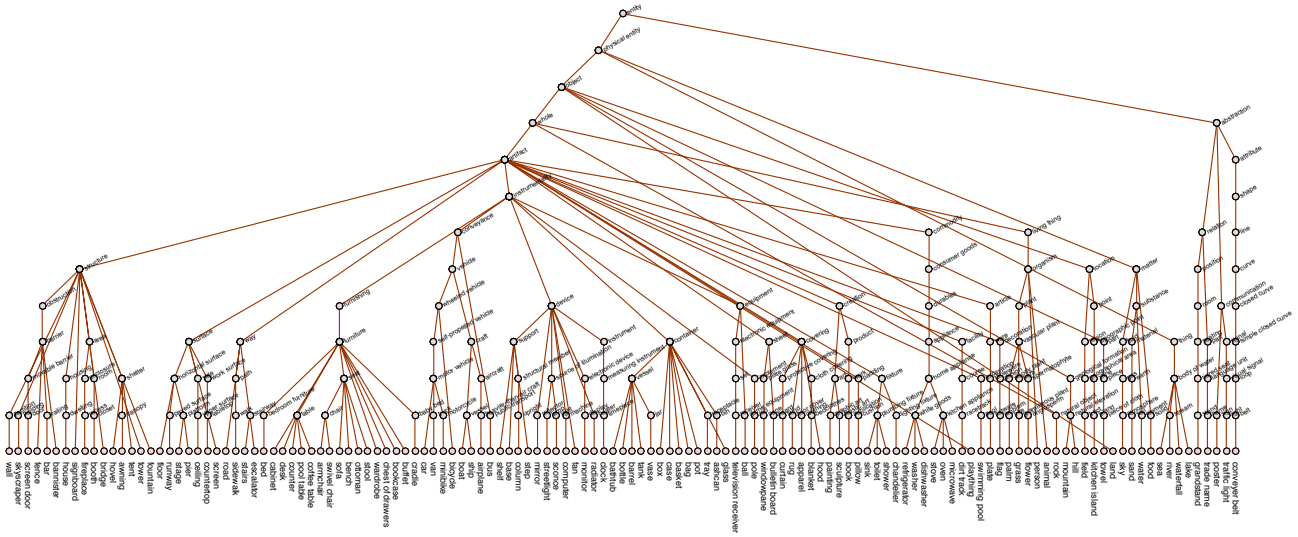


Figure 7. Wordnet tree constructed from the 150 objects in the SceneParse150 benchmark. Clusters inside the wordnet tree represent various hierarchical semantic relations among objects.

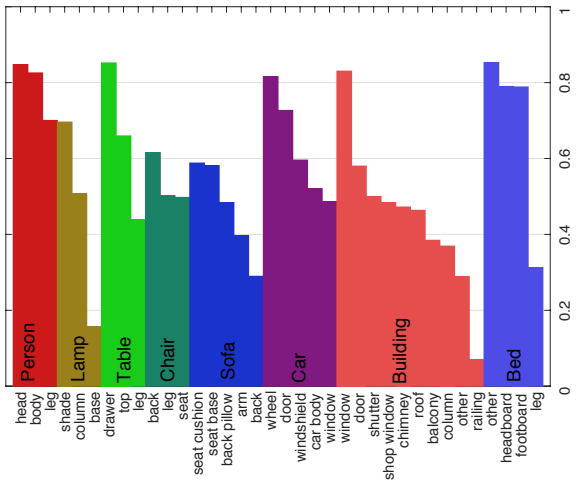


Figure 10. The part segmentation accuracy grouped by the objects.

The results of joint segmentation for stuff, objects, and object parts are shown in Fig. 9. In a single forward pass the network with the proposed cascade module is able to parse scenes at different levels. We use the accuracy instead of the IoU as the metric to measure the part segmentation results, as the parts of object instances in the dataset are not fully annotated. The accuracy for all the parts of the eight objects is plotted in Fig.10. The average accuracy is 55.47%.

4.3. Further applications

Accurate scene parsing leads to wider applications. Here we take the hierarchical semantic segmentation and the automatic scene content removal as exemplar applications of the scene parsing networks.

Hierarchical semantic segmentation. Given the wordnet tree constructed on the object annotation shown in Fig.7, the 150 objects are hierarchically connected and have hyponyms relations. Thus we could gradually merge the objects into their hyponyms so that objects with similar semantics are merged at the early levels. Through this way, we generated a hierarchical semantic segmentation of the image shown in Fig. 11. The tree also provides a principled way to segment more general visual concepts. For example, to detect all furniture in a scene, we can simply merge the hyponyms associated with each synset, such as the chair, table, bench, and bookcase.

Automatic image content removal. Image content removal methods typically require the users to annotate the precise boundary of the target objects to be removed. Here, based on the predicted object probability map from Cascade-SegNet, we automatically identify the image region of the target objects. After cropping out the target objects using the predicted object probability map, we simply use image completion/inpainting methods to fill the holes in the image. Fig. 12 shows some examples of the automatic image content removal. It can be seen that with the predicted object score maps, we are able to crop out the objects from the image in a precise way. We used the image completion technique described in [11].

5. Conclusion

In this paper, we introduced a new densely annotated dataset with the instances of stuff, objects, and parts, covering a diverse set of visual concepts in scenes. The dataset was carefully annotated by a single annotator to ensure precise object boundaries within the image and the consistency of object naming across the images. A generic network de-

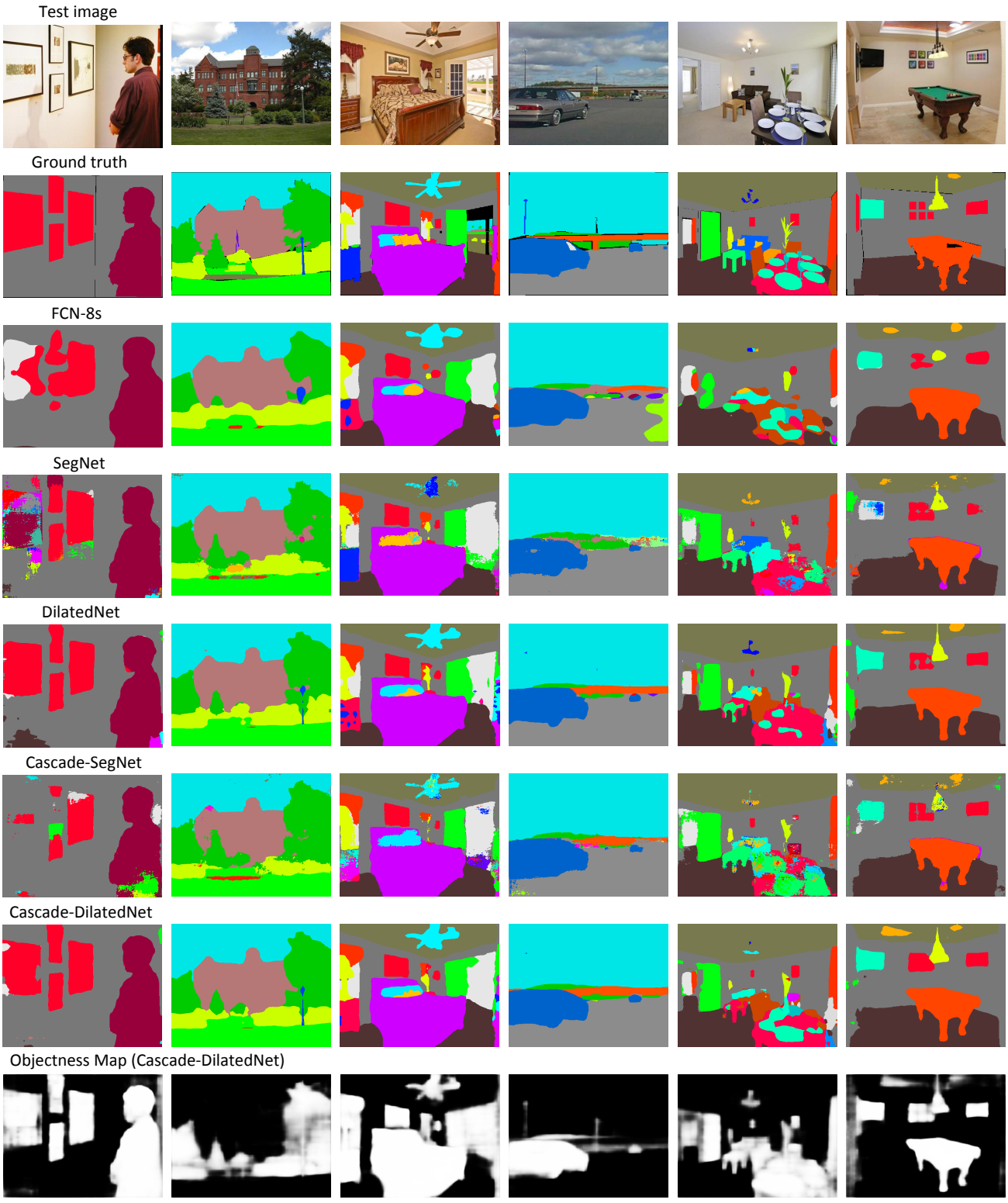


Figure 8. Ground-truths, segmentation results given by the baselines and the cascade networks, and the objectness map and stuff segmentation given by the Cascade-DilatedNet.

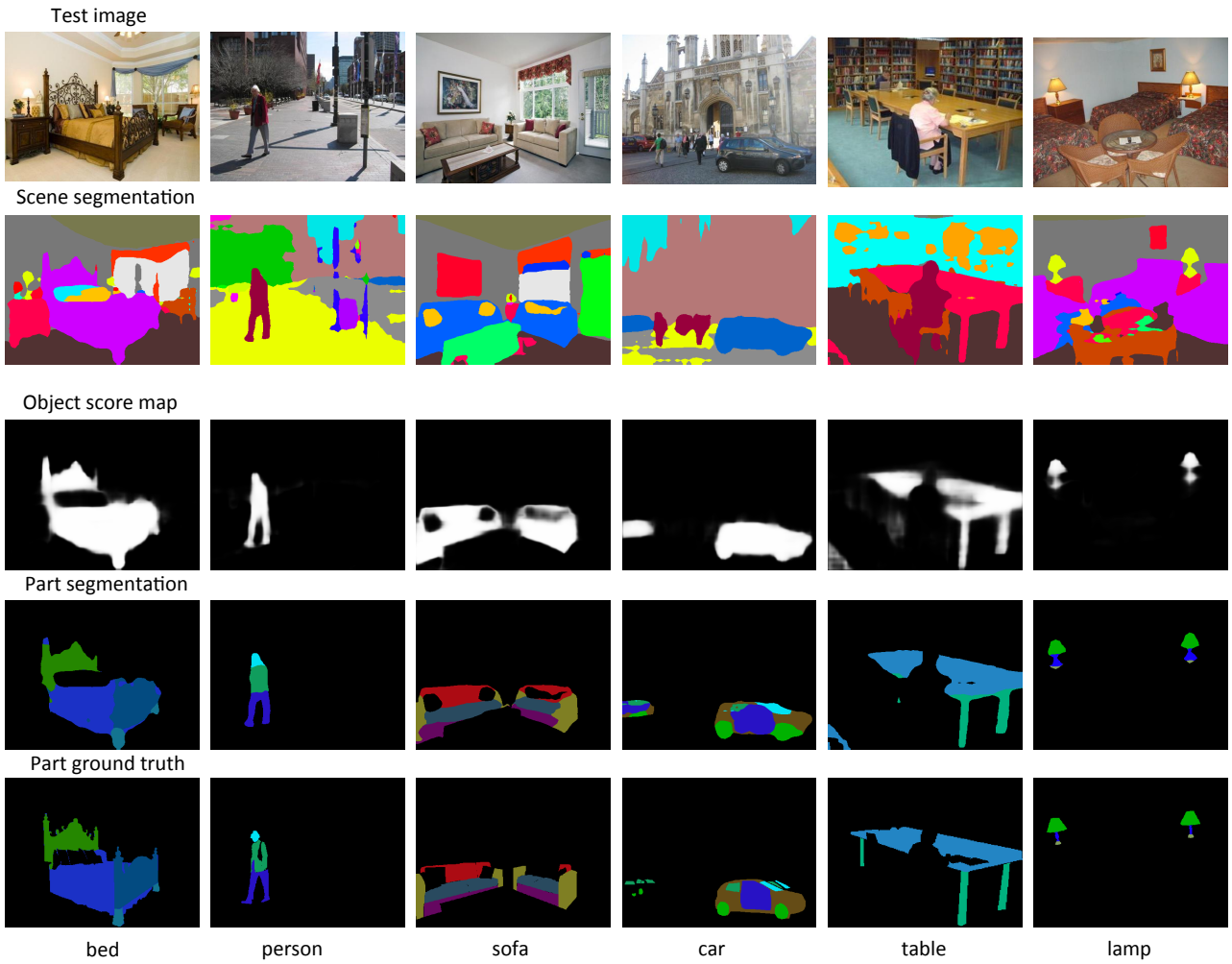


Figure 9. Part segmentation results. The middle row is the object score map predicted by the object stream for each object. The part stream further segments the object score map into different parts.

sign called Cascade Segmentation Module was proposed for scene parsing. It enables the convolutional neural networks to parse scenes into stuff, objects, and object parts in cascade with the state-of-the-art performance.

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv:1511.00561*, 2015.

[2] S. Bell, P. Upchurch, N. Snavely, and K. Bala. OpenSurfaces: A richly annotated catalog of surface appearance. *ACM Trans. on Graphics (SIGGRAPH)*, 32(4), 2013.

[3] S. Bell, P. Upchurch, N. Snavely, and K. Bala. Material recognition in the wild with the materials in context database. In *Proc. CVPR*, 2015.

[4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *arXiv:1606.00915*, 2016.

[5] X. Chen, R. Mottaghi, X. Liu, N.-G. Cho, S. Fidler, R. Urtasun, and A. Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Proc. CVPR*, 2014.

[6] M. Cordts, M. Omran, S. Ramos, T. Scharwächter, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset. In *CVPR Workshop on The Future of Datasets in Vision*, 2015.

[7] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. In *Proc. CVPR*, 2015.

[8] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. *Proc. CVPR*, 2016.

[9] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *Int'l Journal of Computer Vision*, 2010.

[10] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc. CVPR*, 2012.

[11] J.-B. Huang, S. B. Kang, N. Ahuja, and J. Kopf. Image completion using planar structure guidance. *ACM Transactions*

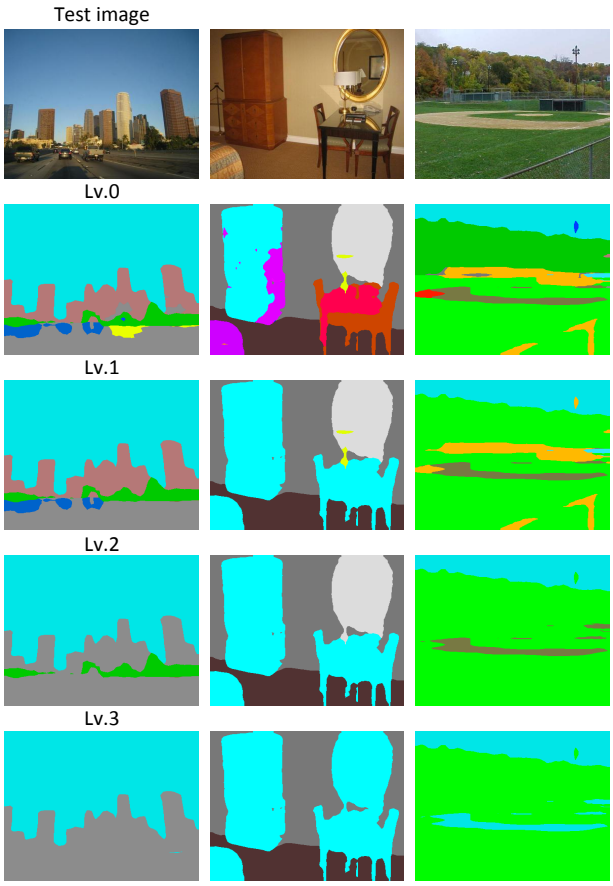


Figure 11. The examples of the hierarchical semantic segmentation. Objects with similar semantics like furnitures and vegetations are merged at early levels following the wordnet tree.

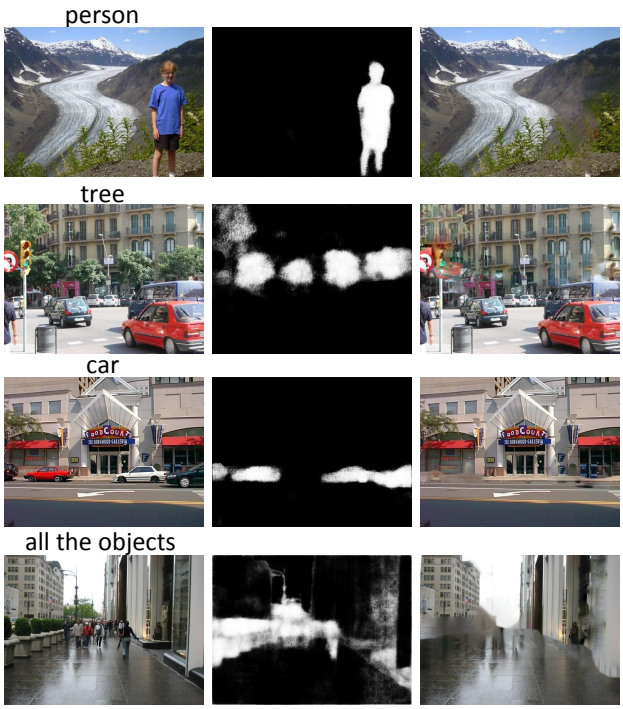


Figure 12. Automatic image content removal using the predicted object score maps given by the scene parsing network. We are not only able to remove individual objects such as person, tree, car, but also groups of them or even all the discrete objects. For each row, the first image is the original image, the second is the object score map, and the third one is the filled-in image.

on Graphics (TOG), 2014.

[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*. ACM, 2014.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012.

[14] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Proc. ECCV*. 2014.

[15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proc. CVPR*, 2015.

[16] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. ICCV*, 2001.

[17] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille. The role of context for object detection and semantic segmentation in the wild. In *Proc. CVPR*, 2014.

[18] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *Proc. ECCV*, 2012.

[19] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *Proc. ICCV*, 2015.

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *Int'l Journal of Computer Vision*, 115(3):211–252, 2015.

[21] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *Int'l Journal of Computer Vision*, 2008.

[22] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proc. CVPR*, 2015.

[23] M. Spain and P. Perona. Measuring and predicting object importance. *International Journal of Computer Vision*, 2010.

[24] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Proc. CVPR*, 2010.

[25] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[26] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *In Advances in Neural Information Processing Systems*, 2014.

Anchoring and Agreement in Syntactic Annotations

Yevgeni Berzak
CSAIL MIT
berzak@mit.edu

Yan Huang
Language Technology Lab
DTAL Cambridge University
yh358@cam.ac.uk

Andrei Barbu
CSAIL MIT
andrei@0xab.com

Anna Korhonen
Language Technology Lab
DTAL Cambridge University
alk23@cam.ac.uk

Boris Katz
CSAIL MIT
boris@mit.edu

Abstract

We present a study on two key characteristics of human syntactic annotations: anchoring and agreement. Anchoring is a well known cognitive bias in human decision making, where judgments are drawn towards pre-existing values. We study the influence of anchoring on a standard approach to creation of syntactic resources where syntactic annotations are obtained via human editing of tagger and parser output. Our experiments demonstrate a clear anchoring effect and reveal unwanted consequences, including overestimation of parsing performance and lower quality of annotations in comparison with human-based annotations. Using sentences from the Penn Treebank WSJ, we also report systematically obtained inter-annotator agreement estimates for English dependency parsing. Our agreement results control for parser bias, and are consequential in that they are *on par* with state of the art parsing performance for English newswire. We discuss the impact of our findings on strategies for future annotation efforts and parser evaluations.[1]

1 Introduction

Research in NLP relies heavily on the availability of human annotations for various linguistic prediction tasks. Such resources are commonly treated as de facto "gold standards" and are used for both training and evaluation of algorithms for automatic annotation. At the same time, human agreement on these annotations provides an indicator for the difficulty of the task, and can be instrumental for estimating upper limits for the performance obtainable by computational methods.

Linguistic gold standards are often constructed using pre-existing annotations, generated by automatic tools. The output of such tools is then manually corrected by human annotators to produce the gold standard. The justification for this annotation methodology was first introduced in a set of experiments on POS tag annotation conducted as part of the Penn Treebank project (Marcus et al., 1993). In this study, the authors concluded that tagger-based annotations are not only much faster to obtain, but also more consistent and of higher quality compared to annotations from scratch. Following the Penn Treebank, syntactic annotation projects for various languages, including German (Brants et al., 2002), French (Abeillé et al., 2003), Arabic (Maamouri et al., 2004) and many others, were annotated using automatic tools as a starting point. Despite the widespread use of this annotation pipeline, there is, to our knowledge, little prior work on syntactic annotation quality and on the reliability of system evaluations on such data.

In this work, we present a systematic study of the influence of automatic tool output on characteristics of annotations created for NLP purposes. Our investigation is motivated by the hypothesis that annotations obtained using such methodologies may be

[1] The experimental data in this study will be made publicly available.

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2215–2224,
Austin, Texas, November 1-5, 2016.

subject to the problem of *anchoring*, a well established and robust cognitive bias in which human decisions are affected by pre-existing values (Tversky and Kahneman, 1974). In the presence of anchors, participants reason relative to the existing values, and as a result may provide different solutions from those they would have reported otherwise. Most commonly, anchoring is manifested as an alignment *towards* the given values.

Focusing on the key NLP tasks of POS tagging and dependency parsing, we demonstrate that the standard approach of obtaining annotations via human correction of automatically generated POS tags and dependencies exhibits a clear anchoring effect – a phenomenon we refer to as *parser bias*. Given this evidence, we examine two potential adverse implications of this effect on parser-based gold standards.

First, we show that parser bias entails substantial overestimation of parser performance. In particular, we demonstrate that bias towards the output of a specific tagger-parser pair leads to over-estimation of the performance of these tools relative to other tools. Moreover, we observe general performance gains for automatic tools relative to their performance on human-based gold standards. Second, we study whether parser bias affects the quality of the resulting gold standards. Extending the experimental setup of Marcus et al. (1993), we demonstrate that parser bias may lead to *lower* annotation quality for parser-based annotations compared to human-based annotations.

Furthermore, we conduct an experiment on inter-annotator agreement for POS tagging and dependency parsing which controls for parser bias. Our experiment on a subset of section 23 of the WSJ Penn Treebank yields agreement rates of 95.65 for POS tagging and 94.17 for dependency parsing. This result is significant in light of the state of the art tagging and parsing performance for English newswire. With parsing reaching the level of human agreement, and tagging surpassing it, a more thorough examination of evaluation resources and evaluation methodologies for these tasks is called for.

To summarize, we present the first study to measure and analyze anchoring in the standard parser-based approach to creation of gold standards for POS tagging and dependency parsing in NLP. We conclude that gold standard annotations that are based on editing output of automatic tools can lead to inaccurate figures in system evaluations and lower annotation quality. Our human agreement experiment, which controls for parser bias, yields agreement rates that are comparable to state of the art automatic tagging and dependency parsing performance, highlighting the need for a more extensive investigation of tagger and parser evaluation in NLP.

2 Experimental Setup

2.1 Annotation Tasks

We examine two standard annotation tasks in NLP, POS tagging and dependency parsing. In the POS tagging task, each word in a sentence has to be categorized with a Penn Treebank POS tag (Santorini, 1990) (henceforth POS). The dependency parsing task consists of providing a sentence with a labeled dependency tree using the Universal Dependencies (UD) formalism (De Marneffe et al., 2014), according to version 1 of the UD English guidelines[2]. To perform this task, the annotator is required to specify the head word index (henceforth HIND) and relation label (henceforth REL) of each word in the sentence.

We distinguish between three variants of these tasks, *annotation*, *reviewing* and *ranking*. In the annotation variant, participants are asked to conduct annotation from scratch. In the reviewing variant, they are asked to provide alternative annotations for all annotation tokens with which they disagree. The participants are not informed about the source of the given annotation, which, depending on the experimental condition can be either parser output or human annotation. In the ranking task, the participants rank several annotation options with respect to their quality. Similarly to the review task, the participants are not given the sources of the different annotation options. Participants performing the annotation, reviewing and ranking tasks are referred to as annotators, reviewers and judges, respectively.

2.2 Annotation Format

All annotation tasks are performed using a CoNLL style text-based template, in which each word appears in a separate line. The first two columns of each line contain the word index and the word, re-

[2] http://universaldependencies.org/#en

spectively. The next three columns are designated for annotation of POS, HIND and REL.

In the annotation task, these values have to be specified by the annotator from scratch. In the review task, participants are required to edit pre-annotated values for a given sentence. The sixth column in the review template contains an additional # sign, whose goal is to prevent reviewers from overlooking and passively approving existing annotations. Corrections are specified following this sign in a space separated format, where each of the existing three annotation tokens is either corrected with an alternative annotation value or approved using a * sign. Approval of all three annotation tokens is marked by removing the # sign. The example below presents a fragment from a sentence used for the reviewing task, in which the reviewer approves the annotations of all the words, with the exception of "help", where the POS is corrected from VB to NN and the relation label *xcomp* is replaced with *dobj*.

```
...
5   you         PRP   6    nsubj
6   need        VBP   3    ccomp
7   help        VB    6    xcomp  # NN * dobj
...
```

The format of the ranking task is exemplified below. The annotation options are presented to the participants in a random order. Participants specify the rank of each annotation token following the vertical bar. In this sentence, the label *cop* is preferred over *aux* for the word "be" and *xcomp* is preferred over *advcl* for the word "Common".

```
...
8   it          PRP   10   nsubjpass
9   is          VBZ   10   auxpass
10  planed      VBN   0    root
11  to          TO    15   mark
12  be          VB    15   aux-cop | 2-1
13  in          IN    15   case
14  Wimbledon   NNP   15   compound
15  Common      NNP   10   advcl-xcomp | 2-1
...
```

The participants used basic validation scripts which checked for typos and proper formatting of the annotations, reviews and rankings.

2.3 Evaluation Metrics

We measure both parsing performance and inter-annotator agreement using tagging and parsing evaluation metrics. This choice allows for a direct comparison between parsing and agreement results. In this context, POS refers to tagging accuracy. We utilize the standard metrics Unlabeled Attachment Score (UAS) and Label Accuracy (LA) to measure accuracy of head attachment and dependency labels. We also utilize the standard parsing metric Labeled Attachment Score (LAS), which takes into account both dependency arcs and dependency labels. In all our parsing and agreement experiments, we exclude punctuation tokens from the evaluation.

2.4 Corpora

We use sentences from two publicly available datasets, covering two different genres. The first corpus, used in the experiments in sections 3 and 4, is the First Certificate in English (FCE) Cambridge Learner Corpus (Yannakoudakis et al., 2011). This dataset contains essays authored by upper-intermediate level English learners[3].

The second corpus is the WSJ part of the Penn Treebank (WSJ PTB) (Marcus et al., 1993). Since its release, this dataset has been the most commonly used resource for training and evaluation of English parsers. Our experiment on inter-annotator agreement in section 5 uses a random subset of the sentences in section 23 of the WSJ PTB, which is traditionally reserved for tagging and parsing evaluation.

2.5 Annotators

We recruited five students at MIT as annotators. Three of the students are linguistics majors and two are engineering majors with linguistics minors. Prior to participating in this study, the annotators completed two months of training. During training, the students attended tutorials, and learned the annotation guidelines for PTB POS tags, UD guidelines, as well as guidelines for annotating challenging syntactic structures arising from grammatical errors. The students also annotated individually six

[3]The annotation bias and quality results reported in sections 3 and 4 use the original learner sentences, which contain grammatical errors. These results were replicated on the error corrected versions of the sentences.

practice batches of 20-30 sentences from the English Web Treebank (EWT) (Silveira et al., 2014) and FCE corpora, and resolved annotation disagreements during group meetings.

Following the training period, the students annotated a treebank of learner English (Berzak et al., 2016) over a period of five months, three of which as a full time job. During this time, the students continued attending weekly meetings in which further annotation challenges were discussed and resolved. The annotation was carried out for sentences from the FCE dataset, where both the original and error corrected versions of each sentence were annotated and reviewed. In the course of the annotation project, each annotator completed approximately 800 sentence annotations, and a similar number of sentence reviews. The annotations and reviews were done in the same format used in this study. With respect to our experiments, the extensive experience of our participants and their prior work as a group strengthen our results, as these characteristics reduce the effect of anchoring biases and increase inter-annotator agreement.

3 Parser Bias

Our first experiment is designed to test whether expert human annotators are biased towards POS tags and dependencies generated by automatic tools. We examine the common out-of-domain annotation scenario, where automatic tools are often trained on an existing treebank in one domain, and used to generate initial annotations to speed-up the creation of a gold standard for a new domain. We use the EWT UD corpus as the existing gold standard, and a sample of the FCE dataset as the new corpus.

Procedure

Our experimental procedure, illustrated in figure 1(a) contains a set of 360 sentences (6,979 tokens) from the FCE, for which we generate three gold standards: one based on human annotations and two based on parser outputs. To this end, for each sentence, we assign *at random* four of the participants to the following annotation and review tasks. The fifth participant is left out to perform the quality ranking task described in section 4.

The first participant annotates the sentence from scratch, and a second participant reviews this annotation. The overall agreement of the reviewers with the annotators is 98.24 POS, 97.16 UAS, 96.3 LA and 94.81 LAS. The next two participants review parser outputs. One participant reviews an annotation generated by the Turbo tagger and parser (Martins et al., 2013). The other participant reviews the output of the Stanford tagger (Toutanova et al., 2003) and RBG parser (Lei et al., 2014). The taggers and parsers were trained on the gold annotations of the EWT UD treebank, version 1.1. Both parsers use predicted POS tags for the FCE sentences.

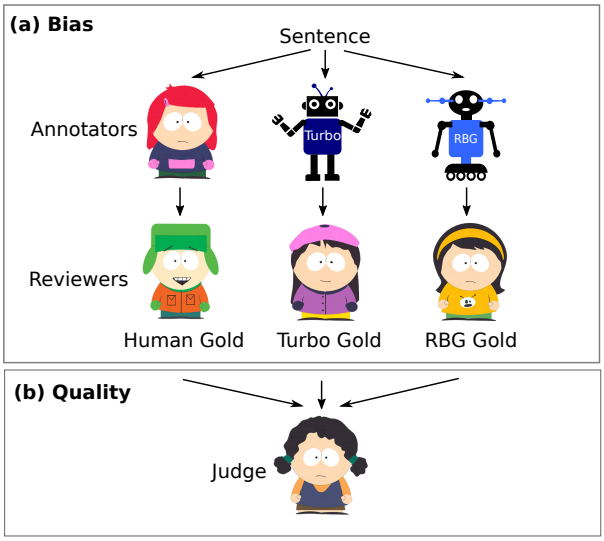


Figure 1: Experimental setup for parser bias (a) and annotation quality (b) on 360 sentences (6,979 tokens) from the FCE. For each sentence, five human annotators are assigned at random to one of three roles: annotation, review or quality assessment. In the bias experiment, presented in section 3, every sentence is annotated by a human, Turbo parser (based on Turbo tagger output) and RBG parser (based on Stanford tagger output). Each annotation is reviewed by a different human participant to produce three gold standards of each sentence: "Human Gold", "Turbo Gold" and "RBG Gold". The fifth annotator performs a quality assessment task described in section 4, which requires to rank the three gold standards in cases of disagreement.

Assigning the reviews to the human annotations yields a human based gold standard for each sentence called "Human Gold". Assigning the reviews to the tagger and parser outputs yields two parser-based gold standards, "Turbo Gold" and "RBG Gold". We chose the Turbo-Turbo and Stanford-RBG tagger-parser pairs as these tools obtain comparable performance on standard evaluation bench-

	Turbo				RBG			
	POS	UAS	LA	LAS	POS	UAS	LA	LAS
Human Gold	95.32	87.29	88.35	82.29	95.59	87.19	88.03	82.05
Turbo Gold	**97.62**	**91.86**	**92.54**	**89.16**	*96.64*	*89.16*	*89.75*	*84.86*
Error Reduction %	49.15	35.96	35.97	38.79	23.81	15.38	14.37	15.65
RBG Gold	96.43	*88.65*	*89.95*	*84.42*	**97.76**	**91.22**	**91.84**	**87.87**
Error Reduction %	23.72	10.7	13.73	12.03	49.21	31.46	31.83	32.42

Table 1: Annotator bias towards taggers and parsers on 360 sentences (6,979 tokens) from the FCE. Tagging and parsing results are reported for the Turbo parser (based on the output of the turbo Tagger) and RBG parser (based on the output of the Stanford tagger) on three gold standards. Human Gold are manual corrections of human annotations. Turbo Gold are manual corrections of the output of Turbo tagger and Turbo parser. RBG Gold are manual corrections of the Stanford tagger and RBG parser. Error reduction rates are reported relative to the results obtained by the two tagger-parser pairs on the Human Gold annotations. Note that (1) The parsers perform equally well on Human Gold. (2) Each parser performs better than the other parser on its own reviews. (3) Each parser performs better on the reviews of the other parser compared to its performance on Human Gold. The differences in (2) and (3) are statistically significant with $p \ll 0.001$ using McNemar's test.

marks, while yielding substantially different annotations due to different training algorithms and feature sets. For our sentences, the agreement between the Turbo tagger and Stanford tagger is 96.97 POS. The agreement between the Turbo parser and RBG parser based on the respective tagger outputs is 90.76 UAS, 91.6 LA and 87.34 LAS.

Parser Specific and Parser Shared Bias

In order to test for parser bias, in table 1 we compare the performance of the Turbo-Turbo and Stanford-RBG tagger-parser pairs on our three gold standards. First, we observe that while these tools perform equally well on Human Gold, each tagger-parser pair performs better than the other on its own reviews. These *parser specific* performance gaps are substantial, with an average of 1.15 POS, 2.63 UAS, 2.34 LA and 3.88 LAS between the two conditions. This result suggests the presence of a bias towards the output of specific tagger-parser combinations. The practical implication of this outcome is that a gold standard created by editing an output of a parser is likely to boost the performance of that parser in evaluations and over-estimate its performance relative to other parsers.

Second, we note that the performance of each of the parsers on the gold standard of the other parser is still higher than its performance on the human gold standard. The average performance gap between these conditions is 1.08 POS, 1.66 UAS, 1.66 LA and 2.47 LAS. This difference suggests an annotation bias towards *shared* aspects in the predictions of taggers and parsers, which differ from the human based annotations. The consequence of this observation is that irrespective of the specific tool that was used to pre-annotate the data, parser-based gold standards are likely to result in higher parsing performance relative to human-based gold standards.

Taken together, the parser specific and parser shared effects lead to a dramatic overall average error reduction of 49.18% POS, 33.71% UAS, 34.9% LA and 35.61% LAS on the parser-based gold standards compared to the human-based gold standard. To the best of our knowledge, these results are the first systematic demonstration of the tendency of the common approach of parser-based creation of gold standards to yield biased annotations and lead to overestimation of tagging and parsing performance.

4 Annotation Quality

In this section we extend our investigation to examine the impact of parser bias on the quality of parser-based gold standards. To this end, we perform a manual comparison between human-based and parser-based gold standards.

Our quality assessment experiment, depicted schematically in figure 1(b), is a ranking task. For each sentence, a randomly chosen judge, who did not annotate or review the given sentence, ranks disagreements between the three gold standards Human Gold, Turbo Gold and RBG Gold, generated in the parser bias experiment in section 3.

Table 2 presents the preference rates of judges

Human Gold Preference %	POS	HIND	REL
Turbo Gold	64.32*	63.96*	61.5*
# disagreements	199	444	439
RBG Gold	56.72	61.38*	57.73*
# disagreements	201	435	440

Table 2: Human preference rates for a human-based gold standard Human Gold over the two parser-based gold standards Turbo Gold and RBG Gold. # disagreements denotes the number of tokens that differ between Human Gold and the respective parser-based gold standard. Statistically significant values for a two-tailed Z test with $p < 0.01$ are marked with *. Note that for both tagger-parser pairs, human judges tend to prefer human-based over parser-based annotations.

for the human-based gold standard over each of the two parser-based gold standards. In all three evaluation categories, human judges tend to prefer the human-based gold standard over both parser-based gold standards. This result demonstrates that the initial reduced quality of the parser outputs compared to human annotations indeed percolates via anchoring to the resulting gold standards.

The analysis of the quality assessment experiment thus far did not distinguish between cases where the two parsers agree and where they disagree. In order to gain further insight into the relation between parser bias and annotation quality, we break down the results reported in table 2 into two cases which relate directly to the *parser specific* and *parser shared* components of the tagging and parsing performance gaps observed in the parser bias results reported in section 3. In the first case, called "parser specific approval", a reviewer approves a parser annotation which disagrees both with the output of the other parser and the Human Gold annotation. In the second case, called "parser shared approval", a reviewer approves a parser output which is shared by both parsers but differs with respect to Human Gold.

Table 3 presents the judge preference rates for the Human-Gold annotations in these two scenarios. We observe that cases in which the parsers disagree are of substantially worse quality compared to human-based annotations. However, in cases of agreement between the parsers, the resulting gold standards do not exhibit a clear disadvantage relative to the Human Gold annotations.

This result highlights the crucial role of parser

Human Gold Preference %	POS	HIND	REL
Turbo specific approval	85.42*	78.69*	80.73*
# disagreements	48	122	109
RBG specific approval	73.81*	77.98*	77.78*
# disagreements	42	109	108
Parser shared approval	51.85	58.49*	51.57
# disagreements	243	424	415

Table 3: Breakdown of the Human preference rates for the human-based gold standard over the parser-based gold standards in table 2, into cases of agreement and disagreement between the two parsers. Parser specific approval are cases in which a parser output approved by the reviewer differs from both the output of the other parser and the Human Gold annotation. Parser shared approval denotes cases where an approved parser output is identical to the output of the other parser but differs from the Human Gold annotation. Statistically significant values for a two-tailed Z test with $p < 0.01$ are marked with *. Note that parser specific approval is substantially more detrimental to the resulting annotation quality compared to parser shared approval.

specific approval in the overall preference of judges towards human-based annotations in table 2. Furthermore, it suggests that annotations on which multiple state of the art parsers agree are of sufficiently high accuracy to be used to save annotation time without substantial impact on the quality of the resulting resource. In section 7 we propose an annotation scheme which leverages this insight.

5 Inter-annotator Agreement

Agreement estimates in NLP are often obtained in annotation setups where both annotators edit the same automatically generated input. However, in such experimental conditions, anchoring can introduce cases of spurious disagreement as well as spurious agreement between annotators due to alignment of one or both participants towards the given input. The initial quality of the provided annotations in combination with the parser bias effect observed in section 3 may influence the resulting agreement estimates. For example, in Marcus et al. (1993) annotators were shown to produce POS tagging agreement of 92.8 on annotation from scratch, compared to 96.5 on reviews of tagger output.

Our goal in this section is to obtain estimates for inter-annotator agreement on POS tagging and dependency parsing that control for parser bias, and

as a result, reflect more accurately human agreement on these tasks. We thus introduce a novel pipeline based on human annotation only, which eliminates parser bias from the agreement measurements. Our experiment extends the human-based annotation study of Marcus et al. (1993) to include also syntactic trees. Importantly, we include an additional review step for the initial annotations, designed to increase the precision of the agreement measurements by reducing the number of errors in the original annotations.

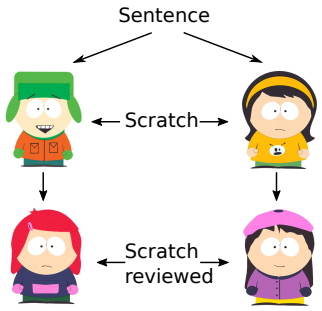


Figure 2: Experimental setup for the inter-annotator agreement experiment. 300 sentences (7,227 tokens) from section 23 of the PTB-WSJ are annotated and reviewed by four participants. The participants are assigned to the following tasks *at random* for each sentence. Two participants annotate the sentence from scratch, and the remaining two participants review one of these annotations each. Agreement is measured on the annotations ("scratch") as well after assigning the review edits ("scratch reviewed").

For this experiment, we use 300 sentences (7,227 tokens) from section 23 of the PTB-WSJ, the standard test set for English parsing in NLP. The experimental setup, depicted graphically in figure 2, includes four participants randomly assigned for each sentence to annotation and review tasks. Two of the participants provide the sentence with annotations from scratch, while the remaining two participants provide reviews. Each reviewer edits one of the annotations independently, allowing for correction of annotation errors while maintaining the independence of the annotation sources. We measure agreement between the initial annotations ("scratch"), as well as the agreement between the reviewed versions of our sentences ("scratch reviewed").

The agreement results for the annotations and the reviews are presented in table 4. The initial agreement rate on POS annotation from scratch is higher than in (Marcus et al., 1993). This difference is likely to arise, at least in part, due to the fact that their experiment was conducted at the beginning of the annotation project, when the annotators had a more limited annotation experience compared to our participants. Overall, we note that the agreement rates from scratch are relatively low. The review round raises the agreement on all the evaluation categories due to elimination of annotation errors present the original annotations.

	POS	UAS	LA	LAS
scratch	94.78	93.07	92.3	88.32
scratch reviewed	**95.65**	**94.17**	**94.04**	**90.33**

Table 4: Inter-annotator agreement on 300 sentences (7,227 tokens) from the PTB-WSJ section 23. "scratch" is agreement on independent annotations from scratch. "scratch reviewed" is agreement on the same sentences after an additional independent review round of the annotations.

Our post-review agreement results are consequential in light of the current state of the art performance on tagging and parsing in NLP. For more than a decade, POS taggers have been achieving over 97% accuracy with the PTB POS tag set on the PTB-WSJ test set. For example, the best model of the Stanford tagger reported in Toutanova et al. (2003) produces an accuracy of 97.24 POS on sections 22-24 of the PTB-WSJ. These accuracies are above the human agreement in our experiment.

With respect to dependency parsing, recent parsers obtain results which are on par or higher than our inter-annotator agreement estimates. For example, Weiss et al. (2015) report 94.26 UAS and Andor et al. (2016) report 94.61 UAS on section 23 of the PTB-WSJ using an automatic conversion of the PTB phrase structure trees to Stanford dependencies (De Marneffe et al., 2006). These results are not fully comparable to ours due to differences in the utilized dependency formalism and the automatic conversion of the annotations. Nonetheless, we believe that the similarities in the tasks and evaluation data are sufficiently strong to indicate that dependency parsing for standard English newswire may be reaching human agreement levels.

6 Related Work

The term "anchoring" was coined in a seminal paper by Tversky and Kahneman (1974), which demonstrated that numerical estimation can be biased by uninformative prior information. Subsequent work across various domains of decision making confirmed the robustness of anchoring using both informative and uninformative anchors (Furnham and Boo, 2011). Pertinent to our study, anchoring biases were also demonstrated when the participants were domain experts, although to a lesser degree than in the early anchoring experiments (Wilson et al., 1996; Mussweiler and Strack, 2000).

Prior work in NLP examined the influence of pre-tagging (Fort and Sagot, 2010) and pre-parsing (Skjærholt, 2013) on human annotations. Our work introduces a systematic study of this topic using a novel experimental framework as well as substantially more sentences and annotators. Differently from these studies, our methodology enables characterizing annotation bias as anchoring and measuring its effect on tagger and parser evaluations.

Our study also extends the POS tagging experiments of Marcus et al. (1993), which compared inter-annotator agreement and annotation quality on manual POS tagging in annotation from scratch and tagger-based review conditions. The first result reported in that study was that tagger-based editing increases inter-annotator agreement compared to annotation from scratch. Our work provides a novel agreement benchmark for POS tagging which reduces annotation errors through a review process while controlling for tagger bias, and obtains agreement measurements for dependency parsing. The second result reported in Marcus et al. (1993) was that tagger-based edits are of higher quality compared to annotations from scratch when evaluated against an additional independent annotation. We modify this experiment by introducing ranking as an alternative mechanism for quality assessment, and adding a review round for human annotations from scratch. Our experiment demonstrates that in this configuration, parser-based annotations are of *lower* quality compared to human-based annotations.

Several estimates of expert inter-annotator agreement for English parsing were previously reported. However, most such evaluations were conducted using annotation setups that can be affected by an anchoring bias (Carroll et al., 1999; Rambow et al., 2002; Silveira et al., 2014). A notable exception is the study of Sampson and Babarczy (2008) who measure agreement on annotation from scratch for English parsing in the SUSANNE framework (Sampson, 1995). The reported results, however, are not directly comparable to ours, due to the use of a substantially different syntactic representation, as well as a different agreement metric. Their study further suggests that despite the high expertise of the annotators, the main source of annotation disagreements was annotation errors. Our work alleviates this issue by using annotation reviews, which reduce the number of erroneous annotations while maintaining the independence of the annotation sources. Experiments on non-expert dependency annotation from scratch were previously reported for French, suggesting low agreement rates (79%) with an expert annotation benchmark (Gerdes, 2013).

7 Discussion

We present a systematic study of the impact of anchoring on POS and dependency annotations used in NLP, demonstrating that annotators exhibit an anchoring bias effect towards the output of automatic annotation tools. This bias leads to an artificial boost of performance figures for the parsers in question and results in lower annotation quality as compared with human-based annotations.

Our analysis demonstrates that despite the adverse effects of parser bias, predictions that are shared across different parsers do not significantly lower the quality of the annotations. This finding gives rise to the following *hybrid annotation* strategy as a potential future alternative to human-based as well as parser-based annotation pipelines. In a hybrid annotation setup, human annotators review annotations on which several parsers agree, and complete the remaining annotations from scratch. Such a strategy would largely maintain the annotation speed-ups of parser-based annotation schemes. At the same time, it is expected to achieve annotation quality comparable to human-based annotation by avoiding parser specific bias, which plays a pivotal role in the reduced quality of single-parser reviewing pipelines.

Further on, we obtain, to the best of our knowl-

edge for the first time, syntactic inter-annotator agreement measurements on WSJ-PTB sentences. Our experimental procedure reduces annotation errors and controls for parser bias. Despite the detailed annotation guidelines, the extensive experience of our annotators, and their prior work as a group, our experiment indicates rather low agreement rates, which are below state of the art tagging performance and on par with state of the art parsing results on this dataset. We note that our results do not necessarily reflect an upper bound on the achievable syntactic inter-annotator agreement for English newswire. Higher agreement rates could in principle be obtained through further annotator training, refinement and revision of annotation guidelines, as well as additional automatic validation tests for the annotations. Nonetheless, we believe that our estimates reliably reflect a realistic scenario of expert syntactic annotation.

The obtained agreement rates call for a more extensive examination of annotator disagreements on parsing and tagging. Recent work in this area has already proposed an analysis of expert annotator disagreements for POS tagging in the absence of annotation guidelines (Plank et al., 2014). Our annotations will enable conducting such studies for annotation with guidelines, and support extending this line of investigation to annotations of syntactic dependencies. As a first step towards this goal, we plan to carry out an in-depth analysis of disagreement in the collected data, characterize the main sources of inconsistent annotation and subsequently formulate further strategies for improving annotation accuracy. We believe that better understanding of human disagreements and their relation to disagreements between humans and parsers will also contribute to advancing evaluation methodologies for POS tagging and syntactic parsing in NLP, an important topic that has received only limited attention thus far (Schwartz et al., 2011; Plank et al., 2015).

Finally, since the release of the Penn Treebank in 1992, it has been serving as the standard benchmark for English parsing evaluation. Over the past few years, improvements in parsing performance on this dataset were obtained in small increments, and are commonly reported without a linguistic analysis of the improved predictions. As dependency parsing performance on English newswire may be reaching human expert agreement, not only new evaluation practices, but also more attention to noisier domains and other languages may be in place.

Acknowledgments

We thank our terrific annotators Sebastian Garza, Jessica Kenney, Lucia Lam, Keiko Sophie Mori and Jing Xian Wang. We are also grateful to Karthik Narasimhan and the anonymous reviewers for valuable feedback on this work. This material is based upon work supported by the Center for Brains, Minds, and Machines (CBMM) funded by NSF STC award CCF-1231216. This work was also supported by AFRL contract No. FA8750-15-C-0010 and by ERC Consolidator Grant LEXICAL (648909).

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. Building a treebank for french. In *Treebanks*, pages 165–187. Springer.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of ACL*, pages 2442–2452.

Yevgeni Berzak, Jessica Kenney, Carolyn Spadine, Jing Xian Wang, Lucia Lam, Keiko Sophie Mori, Sebastian Garza, and Boris Katz. 2016. Universal dependencies for learner english. In *Proceedings of ACL*, pages 737–746.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The tiger treebank. In *Proceedings of the workshop on treebanks and linguistic theories*, volume 168.

John Carroll, Guido Minnen, and Ted Briscoe. 1999. Corpus annotation for parser evaluation. *arXiv preprint cs/9907013*.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Marie-Catherine De Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D Manning. 2014. Universal stanford dependencies: A cross-linguistic typology. In *Proceedings of LREC*, pages 4585–4592.

Karën Fort and Benoît Sagot. 2010. Influence of pre-annotation on pos-tagged corpus development. In *Proceedings of the fourth linguistic annotation workshop*, pages 56–63.

Adrian Furnham and Hua Chu Boo. 2011. A literature review of the anchoring effect. *The Journal of Socio-Economics*, 40(1):35–42.

Kim Gerdes. 2013. Collaborative dependency annotation. *DepLing 2013*, 88.

Tao Lei, Yu Xin, Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. 2014. Low-rank tensors for scoring dependency structures. In *Proceedings of ACL*, volume 1, pages 1381–1391.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The penn arabic treebank: Building a large-scale annotated arabic corpus. In *NEMLAR conference on Arabic language resources and tools*, volume 27, pages 466–467.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

André FT Martins, Miguel Almeida, and Noah A Smith. 2013. Turning on the turbo: Fast third-order non-projective turbo parsers. In *Proceedings of ACL*, pages 617–622.

Thomas Mussweiler and Fritz Strack. 2000. Numeric judgments under uncertainty: The role of knowledge in anchoring. *Journal of Experimental Social Psychology*, 36(5):495–518.

Barbara Plank, Dirk Hovy, and Anders Søgaard. 2014. Linguistically debatable or just plain wrong? In *Proceedings of ACL: Short Papers*, pages 507–511.

Barbara Plank, Héctor Martínez Alonso, Željko Agić, Danijela Merkler, and Anders Søgaard. 2015. Do dependency parsing metrics correlate with human judgments? In *Proceedings of CoNLL*.

Owen Rambow, Cassandre Creswell, Rachel Szekely, Harriet Taber, and Marilyn A Walker. 2002. A dependency treebank for english. In *Proceedings of LREC*.

Geoffrey Sampson and Anna Babarczy. 2008. Definitional and human constraints on structural annotation of english. *Natural Language Engineering*, 14(04):471–494.

Geoffrey Sampson. 1995. English for the computer: Susanne corpus and analytic scheme.

Beatrice Santorini. 1990. Part-of-speech tagging guidelines for the penn treebank project (3rd revision). *Technical Reports (CIS)*.

Roy Schwartz, Omri Abend, Roi Reichart, and Ari Rappoport. 2011. Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation. In *Proceedings of ACL*, pages 663–672.

Natalia Silveira, Timothy Dozat, Marie-Catherine De Marneffe, Samuel R Bowman, Miriam Connor, John Bauer, and Christopher D Manning. 2014. A gold standard dependency corpus for english. In *Proceedings of LREC*, pages 2897–2904.

Arne Skjærholt. 2013. Influence of preprocessing on dependency syntax annotation: speed and agreement. *LAW VII & ID*.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of NAACL*, pages 173–180.

Amos Tversky and Daniel Kahneman. 1974. Judgment under uncertainty: Heuristics and biases. *Science*, 185(4157):1124–1131.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *Proceedings of ACL*, pages 323–333.

Timothy D Wilson, Christopher E Houston, Kathryn M Etling, and Nancy Brekke. 1996. A new look at anchoring effects: basic anchoring and its antecedents. *Journal of Experimental Psychology: General*, 125(4):387.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of ACL*, pages 180–189.