REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
25-03-2023	Final Report	6-Dec-2017 - 21-Aug-2022

4. TITLE AND SUBTITLE
Final Report: GRASP: Global Reading and Assembly for Semantic, Probabilistic World Models

5a. CONTRACT NUMBER
W911NF-18-1-0014

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHORS

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
University of Arizona
PO Box 210158, Rm 510

Tucson, AZ 85721 -0158

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
72292-MI-DRP.1

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Mihai Surdeanu
UU	UU	UU	UU		19b. TELEPHONE NUMBER 520-626-2706

RPPR Final Report
as of 27-Mar-2023

Agency Code: 21XD

Proposal Number: 72292MIDRP **Agreement Number: W911NF-18-1-0014**
INVESTIGATOR(S):

Name: Mihai Surdeanu
Email: msurdeanu@email.arizona.edu
Phone Number: 5206262706
Principal: Y

Organization: **University of Arizona**
Address: PO Box 210158, Rm 510, Tucson, AZ 857210158
Country: USA
DUNS Number: 806345617 EIN: 866004791

Report Date: 21-Sep-2022 Date Received: 25-Mar-2023
Final Report for Period Beginning 06-Dec-2017 and Ending 21-Aug-2022
Title: GRASP: Global Reading and Assembly for Semantic, Probabilistic World Models
Begin Performance Period: 06-Dec-2017 **End Performance Period:** 21-Aug-2022
Report Term: 0-Other

Submitted By: Mihai Surdeanu Email: msurdeanu@email.arizona.edu
Phone: (520) 626-2706

STEM Degrees: **STEM Participants:**

Major Goals: We propose technology that reads and assembles causal, probabilistic models relevant to national and global security analysis. There is an acute need for such methods: currently most such analyses are done manually, taking years to complete. Our technology will enable analysts to quickly build and analyze such models to understand what crises are possible and how to avert them.

Accomplishments: (please see uploaded document)

Training Opportunities: Nothing to Report

Results Dissemination: Several publications were produced by the Arizona team and its subcontracts. Further, multiple open-source software projects were created during this effort. Please see the uploaded report for details.

Honors and Awards: Nothing to Report

Protocol Activity Status:

Technology Transfer: Nothing to Report

PARTICIPANTS:

Participant Type: PD/PI
Participant: Mihai Surdeanu
Person Months Worked: 1.00 **Funding Support:**
Project Contribution:
National Academy Member: N

Participant Type: Co PD/PI
Participant: Clayton Morrison
Person Months Worked: 1.00 **Funding Support:**
Project Contribution:

National Academy Member: N

Participant Type: Co-Investigator
Participant: Steven Bethard
Person Months Worked: 1.00 **Funding Support:**
Project Contribution:
National Academy Member: N

Participant Type: Co PD/PI
Participant: Benjamin Gyori
Person Months Worked: 1.00 **Funding Support:**
Project Contribution:
National Academy Member: N

Participant Type: Co-Investigator
Participant: Adarsh Pyarelal
Person Months Worked: 1.00 **Funding Support:**
Project Contribution:
National Academy Member: N

Participant Type: Co PD/PI
Participant: Gerrit Hoogenboom
Person Months Worked: 1.00 **Funding Support:**
Project Contribution:
National Academy Member: N

Participant Type: Co PD/PI
Participant: Pascale Proux
Person Months Worked: 1.00 **Funding Support:**
Project Contribution:
National Academy Member: N

RPPR Final Report
as of 27-Mar-2023

Partners

,

I certify that the information in the report is complete and accurate:
Signature: Mihai Surdeanu
Signature Date: 3/25/23 3:35PM

World Modelers Final Report for the Arizona Team

Table of Contents

1. Foreword

This document summarizes the scientific progress of the Arizona team in the World Modelers program. Note that, despite the name, the Arizona team includes several groups in addition to the University of Arizona group. In particular, this contract included the Harvard Medical School (Benjamin Gyori lead), University of Florida (Gerrit Hoogenboom lead), and Uncharted (Bill Wright and Pascale Proux leads).

The research areas investigated impacted most of the relevant areas of the World Modelers program: machine reading, assembly, context extraction, top-down modeling, bottom-up modeling, human-machine interfaces and visualization. The contributions to all these areas are detailed throughout the report.

Since this is the final report for our participation in the World Modelers program, we would like to express our deep appreciation for the opportunity to participate in this DARPA program. We would like to thank DARPA, as well as Joshua Elliott, the program manager.

2. Machine Reading

2.1. Eidos: Machine Reading for Causality

The first key contribution of the machine reading team is the development of Eidos, a rule-based open-domain IE system that extracts causal statements from raw text. To maximize domain independence, Eidos is largely unlexicalized (with the exception of causal cues such as *promotes*), and implements a top-down approach where causal interactions are extracted first, followed by the participating concepts, which are grounded with specific geospatial and temporal contexts for model contextualization. Eidos also extracts quantifiable adjectives (e.g. *significant*) that can be used to form a bridge between qualitative statements and quantitative modeling.

To understand the top-down approach in Eidos, let us consider the individual steps involved in processing the following sentence: *The significantly increased conflict seen in South Sudan forced many families to flee in 2017.*

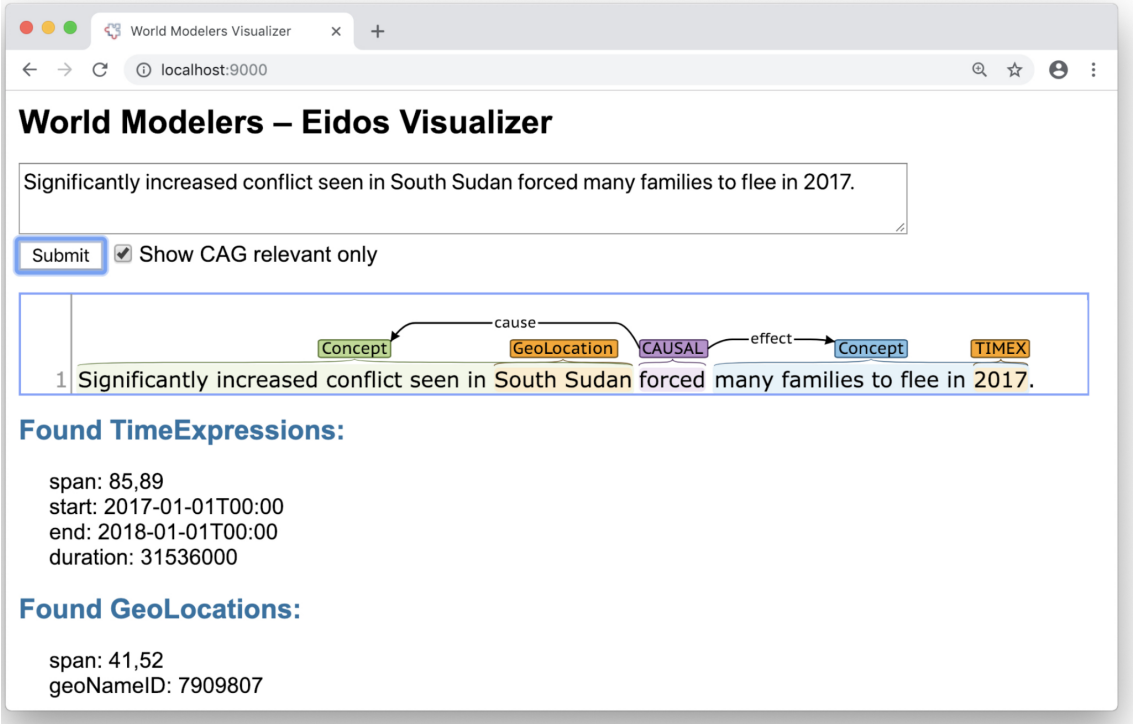
(1) We begin by preprocessing the text with dependency syntax using Stanford CoreNLP (Manning et al., 2014) and the **processors** library (https://github.com/clulab/processors).

(2) Then, Eidos finds any occurrences of quantifiers (gradable adjectives and adverbs). These are common in the high-level texts relevant to food insecurity, such as reports from UN agencies and nonprofits, but they are difficult to use in quantitative models without additional information. In the example above, the word *significantly* is found as a quantifier of *increased*. Delphi (see section 5) uses these quantifiers to construct probability density functions using the crowdsourced data of Sharp et al. (2018).

(3) Next, Eidos uses a set of trigger words to find causal and correlational relations with an Odin grammar (Valenzuela-Escarcega et al., 2016). Odin is an information extraction framework which includes a declarative language supporting both surface and syntactic patterns and a runtime system. Eidos's grammar was based in part on the biomedical grammar developed by Valenzuela-Escarcega et al. (2018) but adapted to the open domain and our representation of concepts. This rule grammar is fully interpretable and easily editable, allowing users to make modifications without needing to retrain a complex model. In the example sentence from earlier, the extraction of a causal relation would be triggered by the word *forced*, with *conflict* and *families* identified as the initial cause and effect, respectively.

(4) The initial cause and effect are then expanded using dependency syntax following the approach of Hahn-Powell et al. (2017). Namely, from each of the initial arguments, we traverse outgoing dependency links to expand the arguments into their dependency subgraph. Here, the resulting arguments are *significantly increased conflict seen in South Sudan and many families to flee in 2017.*

(5) Relevant state information is then added to the expanded concepts. Representing the polarity of an influence on the causal relation edge (i.e., in terms of *promotes* or *inhibits*) can be lossy, so Eidos instead uses concept states (i.e., concepts can be increased, decreased, and/or quantified). In the example above, Eidos marks the concept pertaining to conflict as being increased and quantified. If desired, the promotion/inhibition representation with edge polarity can be straightforwardly recovered. The final output of the Eidos system for the running example sentence, as displayed in the Eidos webapp, is shown in the figure below.



Screenshot of the Eidos output for the running example sentence, visualized in Eidos's webapp.

2.1.1. Concept linking

The Eidos reading system, with its top-down approach, was designed to keep extracted concepts as close to the text as possible, intentionally allowing downstream users to make decisions about event semantics depending on their use cases. As a result, linking concepts to a taxonomy becomes critical for preventing sparsity. Eidos's concept linking is based on word embedding similarities, coupled with a compositional algorithm that accounts for the subject-verb-object structure of the event. In particular, a given concept to be grounded is represented as a 4-tuple, where the slots represent (in order):

- the theme of the concept,
- any property of that theme,
- the process that applies to or acts on the theme, if any, and
- any property of that process.

Each of the tuple elements is encoded using word embeddings and aligned with relevant taxonomy nodes, e.g., the process text must be grounded to a taxonomy element that inherits from Process, while the theme is grounded to either a Concept sub-node or Process one.

In practice, Eidos returns the top *k* groundings, allowing for downstream disambiguation. The concept linking strategy is modular and allows for grounding to any taxonomy provided in the human-readable YAML format. With this method, Eidos is able to link to an arbitrary number of taxonomies, at both high and low levels of abstraction.

2.1.2. Temporal and geospatial normalization

Time normalization. The context surrounding the extractions is often critical for downstream reasoning. Eidos integrates the temporal parser of Laparra et al. (2018) that uses a character recurrent neural network to identify time expressions in the text which are then linked together with a set of rules into semantic graphs which follow the SCATE schema (Bethard and Parker, 2016) and can be interpreted using temporal logic to obtain the intervals referred to by the time expressions. After the time expressions are identified and normalized, an Odin grammar attaches them to the causal relations extracted by Eidos. If the document creation time is provided, it is also parsed by our model and used as the default temporal attachment for those causal relations without a temporal expression in their close context.

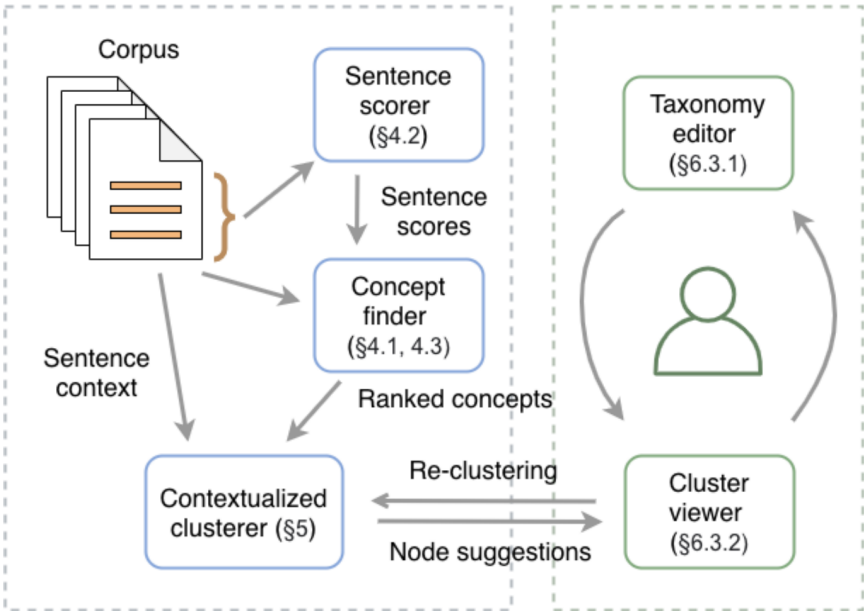
Geospatial normalization. Eidos's geospatial normalization module (Yadav et al., 2019) has two components: a detection component consisting of the word-level LSTM named entity recognition (NER) model of Yadav and Bethard (2018), and a normalization component which implements population heuristics (i.e., selecting the most populous location (Magge et al., 2018)) and filters using a distance-based heuristic (Magge et al., 2018).

Please see Section 4 for more details on the extraction of temporal and spatial context.
For more details on Eidos please see the Eidos Wiki: https://github.com/clulab/eidos/wiki.

2.2. Taxonomy Builder

The second key contribution of the machine reading team is contributing to the development of Taxonomy Builder, a tool that is designed to streamline the taxonomy acquisition process. At a high level, this is achieved by using automatically gleaned text summarization analytics from the corpus itself, coupled with the power and expressivity of recent contextualized embeddings, to suggest candidate concepts to a human user during an interactive session. The user can accept, reject, or manipulate the suggested concept, then determine where it belongs in relation to other existing nodes. Taxonomies can be persisted for downstream use or a subsequent editing session.

In more detail, our approach for rapid data-driven taxonomy generation combines insights from several layers of text understanding to suggest highly relevant candidate concepts to a user, who decides how to use them (or not) to build the taxonomy they need. The architecture is shown in the figure below.



Overall architecture of Taxonomy Builder, which combines offline pre-processing steps (the components in the left panel) with online user sessions (right panel).

Specifically, given a corpus of documents and optionally an existing taxonomy, we assign salience scores to each sentence based on keyword occurrence. From sentences with a sufficiently high salience, we extract multi-word expressions (noun and verb phrases) as potential phrases of interest. These are ranked with respect to each other using an extension of the TextRank algorithm (Mihalcea and Tarau, 2004). The top-ranked concepts are encoded with contextualized word embeddings and clustered. Resulting clusters are presented to the user as suggested novel concepts; the phrases in the clusters can be thought of as examples of that concept. To ensure interactivity, these computationally expensive steps are done ahead of time, and the user need only load the pre-computed initial clusters.

Once loaded, clusters are exposed to the user through an interactive web application. The user can then decide between a series of actions; specifically, they can *accept*, *edit*, *skip*, or *discard* the node. Accepted nodes are then inserted into the current working taxonomy. The user continues to work through the system's suggestions until they are satisfied with the taxonomy.

For more details please see the paper that describes the complete system in detail (Hungerford et al., 2022). The software is freely available here: https://github.com/twosixlabs-dart/dart-ui.

2.3. Other Machine Reading Contributions

In addition to the above contributions, the machine reading team contributed several other components to the World Modelers ecosystem. We briefly summarize them below:

Concept aligner: The team has extended the grounding component from Eidos (see Section 2.1.1) to link concepts extracted by machine reading from text with indicators in databases used by bottom-up models.

Processors: This is a Scala library that implements modern neural approaches for core NLP applications such as part-of-speech tagging, named entity recognition, syntactic parsing, semantic role labeling, etc. The focus is on models that are light and practical such that this library runs well on commodity hardware. The software is freely available at: https://github.com/clulab/processors.

Explainable NLP: This effort (Tang and Surdeanu, 2022) introduces an explainable approach for relation extraction that mitigates the tension between generalization and explainability by jointly training for the two goals. Our approach uses a multi-task learning architecture, which jointly trains a classifier for relation extraction, and a sequence model that labels words in the context of the relations that explain the decisions of the relation classifier. We also convert the model outputs to rules to bring global explanations to this approach. This sequence model is trained using a hybrid strategy: supervised, when supervision from pre-existing patterns is available, and semi-supervised otherwise. In the latter situation, we treat the sequence model's labels as latent variables, and learn the best assignment that maximizes the performance of the relation classifier. We evaluate the proposed approach on the two datasets and show that the sequence model provides labels that serve as accurate explanations for the relation classifier's decisions, and, importantly, that the joint training generally improves the performance of the relation classifier. We also evaluate the performance of the generated rules and show that the new rules are a great add-on to the manual rules and bring the rule-based system much closer to the neural models.

2.4. Machine Reading References

Eidos, INDRA, & Delphi: From Free Text to Executable Causal Models. Rebecca Sharp; Adarsh Pyarelal; Benjamin Gyori; Keith Alcock; Egoitz Laparra; Marco A. Valenzuela-Escarcega; Ajay Nagesh; Vikas Yadav; John Bachman; Zheng Tang; Heather Lent; Fan Luo; Mithun Paul; Steven Bethard; Kobus Barnard; Clayton Morrison; and Mihai Surdeanu. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations), pages 42-47, Minneapolis, Minnesota, 6 2019.

Grounding Gradable Adjectives through Crowdsourcing. Rebecca Sharp; Mithun Paul; Ajay Nagesh; Dane Bell; and Mihai Surdeanu. In LREC 2018, 2018.

Large-scale Automated Machine Reading Discovers New Cancer Driving Mechanisms. Marco A. Valenzuela-Escarcega; Ozgun Babur; Gus Hahn-Powell; Dane Bell; Thomas Hicks; Enrique Noriega-Atala; Xia Wang; Mihai Surdeanu; Emek Demir; and Clayton T. Morrison. Database: The Journal of Biological Databases and Curation. 2018.

Marco A. Valenzuela-Escarcega, Gustave Hahn-Powell, and Mihai Surdeanu. 2016. Odin's runes: A rule language for information extraction. In Proceedings of the 10th edition of the Language Resources and Evaluation Conference (LREC).

Rebecca Sharp, Mithun Paul, Ajay Nagesh, Dane Bell, and Mihai Surdeanu. 2018. Grounding gradable adjectives through crowdsourcing. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Paris, France. European Language Resources Association (ELRA).

Egoitz Laparra, Dongfang Xu, and Steven Bethard. 2018. From characters to time intervals: New paradigms for evaluation and neural parsing of time normalizations. Transactions of the Association for Computational Linguistics, 6:343–356.

Steven Bethard and Jonathan Parker. 2016. A semantically compositional annotation scheme for time normalization. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), Paris, France. European Language Resources Association (ELRA).

Vikas Yadav, Egoitz Laparra, Ti-Tai Wang, Mihai Surdeanu, and Steven Bethard. 2019. University of arizona at semeval-2019 task 12: Deep-affix named entity recognition of geolocation entities. In Proceedings of The 13th International Workshop on Semantic Evaluation, Minneapolis, USA. Association for Computational Linguistics.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In Proceedings of the 27th International Conference on Computational Linguistics, pages 2145–2158.

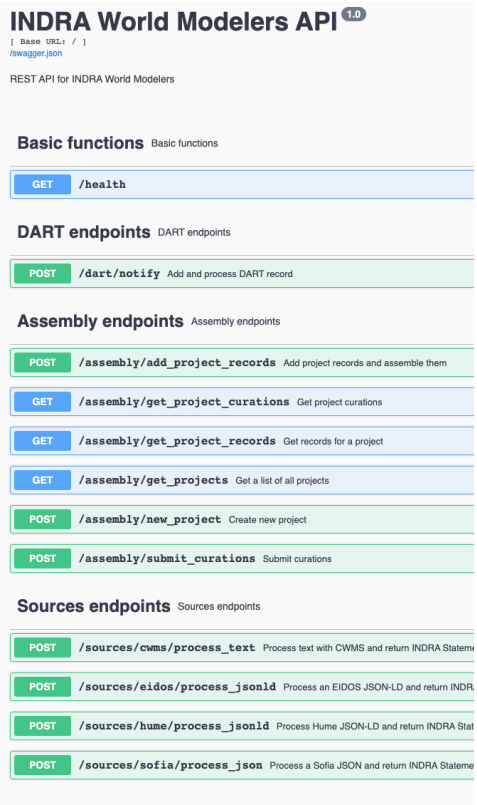
Arjun Magge, Davy Weissenbacher, Abeed Sarker, Matthew Scotch, and Graciela Gonzalez-Hernandez. 2018. Deep neural networks and distant supervision for geographic location mention extraction. Bioinformatics, 34(13):i565–i573.

Taxonomy Builder: a Data-driven and User-centric Tool for Streamlining Taxonomy Construction. John Hungerford; Yee Seng Chan; Jessica MacBride; Benjamin M. Gyori; Andrew Zupon; Zheng Tang; Egoitz Laparra; Haoling Qiu; Bonan Min; Yan Zverev; Caitlin Hilverman; Max Thomas; Walt Andrews; Keith Alcock; Zeyu Zhang; Michael Reynolds; Mihai Surdeanu; Steve Bethard; and Rebecca Sharp. In Proceedings of the Second Workshop on Bridging Human-Computer Interaction and Natural Language Processing, 2022.

It Takes Two Flints to Make a Fire: Multitask Learning of Neural Relation and Explanation Classifiers. Zheng Tang; and Mihai Surdeanu. Computational Linguistics,1-40. 09 2022.

3. Assembly

The Harvard Medical School team developed INDRA World (https://github.com/indralab/indra_world). INDRA World is a generalization of the INDRA knowledge assembly system for automatically collecting, assembling, and modeling the web of causal relations that drive interconnected events in regional and global systems. INDRA World interfaces with four machine reading systems (including Eidos, introduced in section 2.1 above) that extract concepts, events, and causal relations from text (typically reports from governmental and non-governmental organizations, news stories, and scientific publications). The extractions are processed into a standardized Statement representation and then processed (filtered, normalized, etc.). INDRA World implements assembly logic to find relationships between statements, including matching, contradiction, and refinement (i.e., one statement is a more general or more specific version of the other). It then calculates a belief score which is based on all available evidence directly or indirectly supporting a given statement. INDRA World also implements a database and service architecture to be run as a service that integrates with other systems and supports managing project-specific statement sets and incremental assembly with new reader outputs.

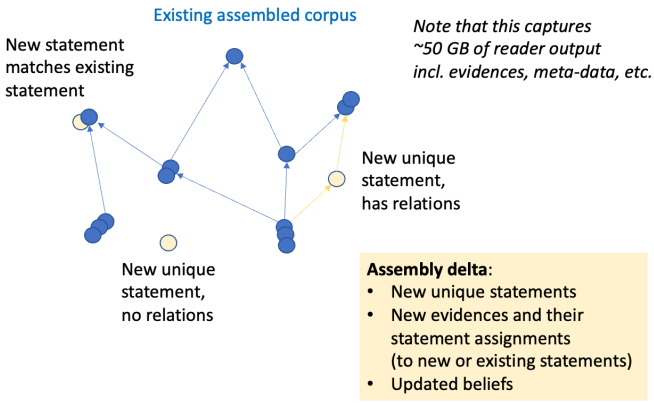


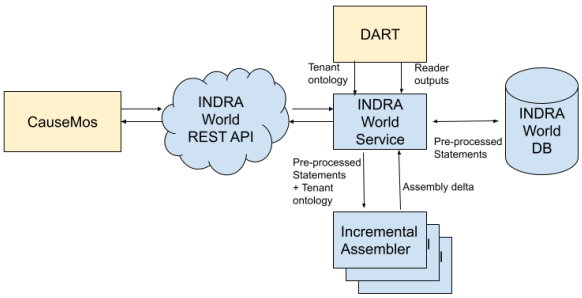
Representation. Causal influences are represented as Statement objects with a cause and effect argument which are Events. An Event represents a core Concept, as well as the polarity and magnitude of a change in that Concept, along with time and geospatial context. This allows representing nuanced causal influences such as the one implied by the following sentence: “A small decrease in subsidies in 2019 in South Sudan resulted in a large increase in agricultural production in 2020.” Each Statement also carries one or more Evidence objects that support it. Each Evidence captures, for instance, the doument and sentence from which the causal influence was extracted, and the reading system by which it was extended.

Importantly, each Concept represented by INDRA World is grounded. To be able to express complex concepts, we use a compositional grounding approach where a Concept is represented as a tuple of a core term, and an optional process, property, and process property. This allows expressing concepts such as “the price of water trucking”

Multi-reader ontology-guided knowledge assembly. INDRA World implements an approach to multi-reader assembly which relies on standardizing representations across diverse outputs from multiple systems that extract causal influences from text. It is also able to normalize across equivalent or partially overlapping causal influences across extractions from multiple documents or sentences. It does this with respect to an ontology to which Concepts are compositionally grounded. For example, if “rain” is a form of “precipitation” according to the input ontology, INDRA World will align Statements about these concepts in a way that this refinement is taken into account. To assess confidence, INDRA uses quantitative measures of grounding accuracy and calculates a relation-level belief score based on direct and indirect evidence supporting a given Statement. The end result is a causal analysis graph that retains high-confidence causal influences along with evidence and metadata.

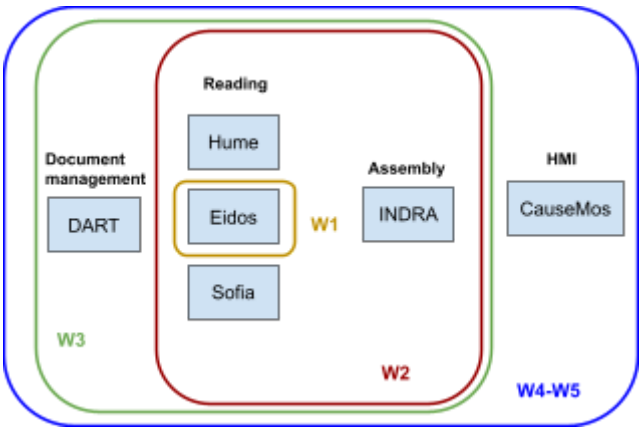
Incremental assembly. Outputs from reading systems on the scale of tens of thousands of documents can be large, encompassing tens of thousands of unique causal influences and on the order of tens of gigabytes of evidence and metadata information. If a small number of new documents are added, this would naively require reloading the entire corpus and re-running assembly. To solve this, we implemented a service-oriented incremental assembly approach centered around a relational database.





The key idea is to recognize Statement equivalence via hashing of the Statement’s content, store associated Evidences in a separate table keyed by Statement hash (allowing incremental extension), and implementing an optimized algorithm for finding partial refinement with respect to a ontology. This allows INDRA World to produce an “assembly delta” with respect to a large existing assembly and a set of new causal influences.

Integrated services. The HMS team also led the development of a service architecture that implements incremental assembly across performers in an integrated, HMI-based system. The user uploads one or more documents through the HMI which get added to a document store. The document store sends out requests to reading systems (running as services) to produce output for the new documents. INDRA World is then notified about the new reader outputs and updates its database. Finally, the user asks for an updated assembly which INDRA produces as an assembly delta displayed by the HMI.



Finally, we created multiple user-instantiatable integrations of the technologies developed for causal knowledge extraction and assembly. Subsystems are containerized and can be run in increasingly complex setups (W1-5) depending on the use case. This approach is documented at https://worldmodelers.com/reading-assembly.html.

The causal graph assembled by INDRA is the basis of knowledge exploration and modeling through the HMI. Modeling and simulation engines such as Delphi and DySE use INDRA-generated relations (in some cases combined with manually asserted relations) as the basis for dynamical modeling of causal analysis graphs.

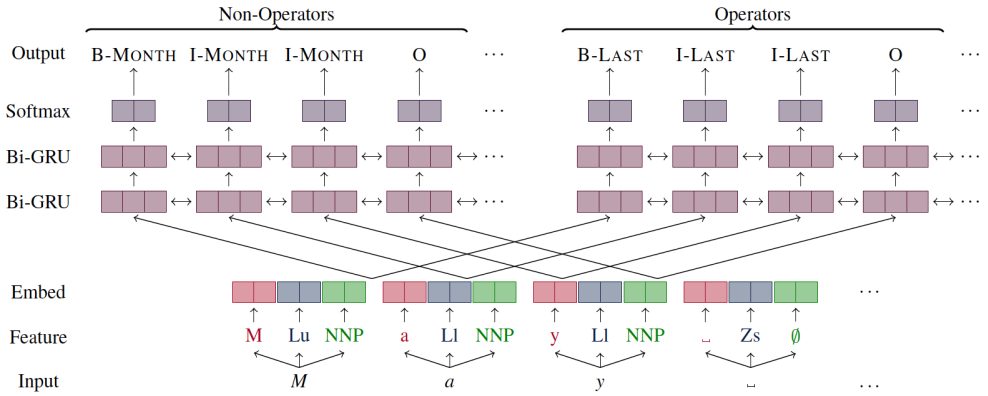
4. Context Extraction

Our main contributions in the areas of context extraction were in identifying and normalizing time and geographical expressions. Our work included approaches to address some of the unique challenges of the World Modelers domains, including recognizing and normalizing new season types (“rainy season”, “rice harvest”, etc.) and unnamed locations described in terms of named locations (”surrounding areas in Upper Nile”).

4.1. Neural time normalization system

We introduced a neural network approach for detecting time expressions and normalizing them to intervals on the timeline, e.g., converting “Last March” in a document from 2017 to [2017-03-01, 2017-04-01). The neural network was trained using data annotated in our prior work (Bethard and Parker 2016) to predict for each character in a text, whether it represented an instance of a temporal entity or temporal operator such as MONTH-OF-YEAR or BETWEEN. As shown below, the neural network included in the representation of each character its unicode character type and the part-of-speech of its containing word, and then used two layers of gated recurrent unit (GRU) networks to integrate context across characters in the text before predicting a temporal type. Our initial version of this model was published in the Transactions of the ACL (Laparra, Xu, and Bethard 2018). Over the course of the project, we improved the performance of this model by 20% by replacing its randomly initialized character embeddings

with contextualized character embeddings pre-trained on a 1-billion word corpus. The improved model was published in the Joint Conference on Lexical and Computational Semantics (Xu, Laparra, and Bethard 2019).



We developed a rule-based system to compose temporal entities and operators that constrained composition based on the type constraints of the temporal operators and on the proximity of the entities and operators being composed. The composed entities and operators could then be interpreted using the temporal operator interpretation library developed in our prior work, e.g.,

```
scala> // the 3-year period following the year 1985
scala> val x = NextP(Year(1985), SimplePeriod(YEARS, 3))
scala> (x.start, x.end)
res0: (java.time.LocalDateTime, java.time.LocalDateTime) =
(1986-01-01T00:00,1989-01-01T00:00)
```

This time normalization system was integrated into Eidos as a Finder, using tensorflow-java to load the neural network model, applying custom Odin rules to attach normalized times to causal entities, and attaching the document creation time to entities lacking explicit times. Over the course of the project, we sped up this system via batching predictions, using regular expressions to filter unlikely-to-be-temporal sentences, and reducing the context considered by the neural network based on analysis of its sensitivity.

4.2. Rule-based time normalization system

As a faster alternative to the neural network based time normalization system, we also developed a rule-based system inspired by our prior work in rule-based time normalization (Bethard 2013). Odin rules were designed to capture different types of time expressions, and the jollyday library was integrated to normalize named holidays like Christmas. An example Odin rule looks like:

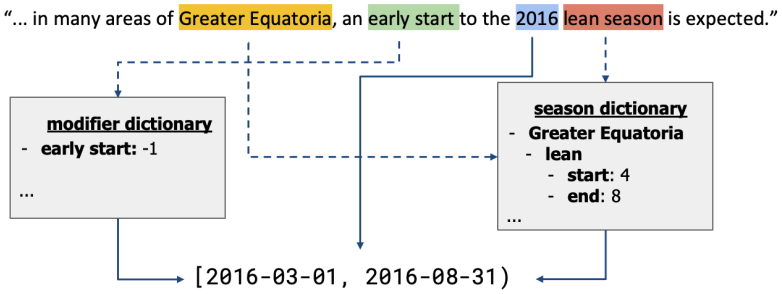
```
- name: date-range-5
  priority: ${rulepriority}
  label: DateRange
```

```
type: token
example: "It took place between May and June 2020"
action: mkDateRangeMentionWithMonth
pattern: |
  /(?i)between/ @month1:PossibleMonth /(?i)(and|to)/
  @month2:PossibleMonth @year:PossibleYear
```

We created an evaluation dataset for time normalization by double-annotating and adjudicating time expressions in 17 World Modelers documents. In our final evaluation, this approach achieved 0.855 f1 on the evaluation corpus.

4.3. Rule-based season normalization system

We designed a rule-based system to normalize seasons like “rainy season” that require knowledge of both time and location, since the timeline intervals of such seasons vary from region to region. To support such a system, we developed a database of regions and the time frames of their seasons, as well as temporal definitions for modifiers like “early start”, in consultation with experts on seasons in Ethiopia and surrounding areas. Odin rules were then designed to detect season names (e.g., “lean season”) and combine them with modifiers and nearby locations (e.g., “Greater Equatoria”) to allow seasons to be normalized to intervals on the timeline.



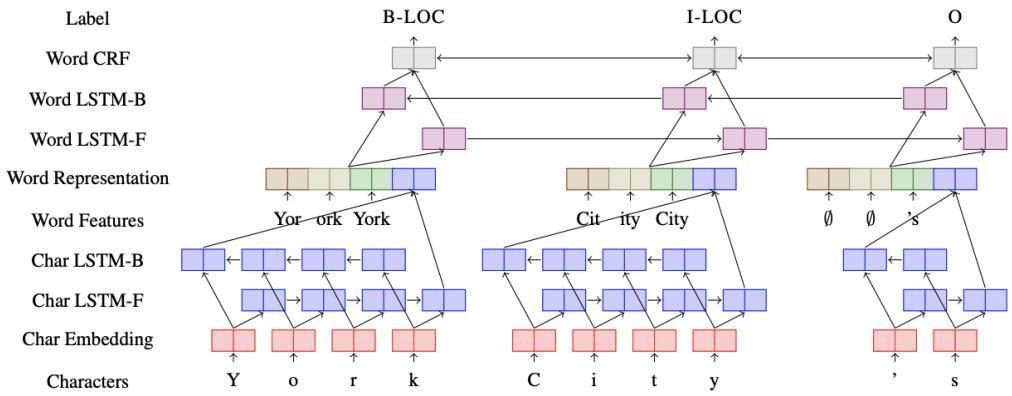
Most frequent “* season” from “Twelve month evaluation” documents:

- 276 lean season
- 210 main season
- 145 second season
- 131 rainy season
- 125 early season
- 108 first season
- 100 dry season
- 69 secondary season
- 68 growing season
- 62 rainfall season

4.4. Geographic name extraction and normalization system

We designed a system for identifying place names and normalizing them to the GeoNames ontology. A neural network processes the text word by word, incorporating character, word and affix information, and predicting for each word whether or not it represents a geographic location. This approach was based on our prior work (Yadav, Sharp, and Bethard 2018) and was

also our submission to the SemEval-2019 Task 12 shared task on geographic location detection and normalization (Yadav et al. 2019).

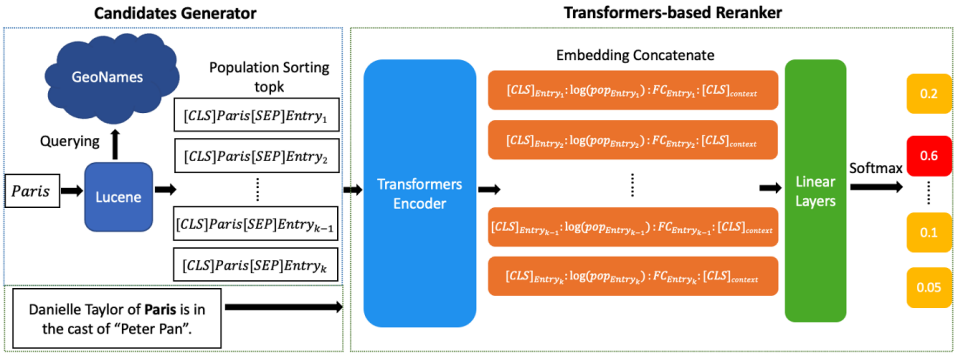


To normalize the locations detected by this system to the GeoNames ontology, we developed a Lucene index of GeoNames and a sieve-based search strategy that applied matching strategies in order from high precision to low precision:

1. The mention exactly matches (ignoring whitespace) the name from the ontology
2. The mention is within a 2 character Levenshtein edit distance (ignoring whitespace) of the name from the ontology
3. The mention has at least one character 3-gram overlap with the name from the ontology
4. The mention has at least one token (according to the Lucene StandardAnalyzer) overlap with the name from the ontology
5. The mention exactly matches the capital letters of the name from the ontology
6. The name from the ontology is a country and the mention exactly matches its country code

We deployed this GeoNames index to Maven Central, and also identified woredas in GADM that were missing from GeoNames, and provided a GeoNames-formatted resource that allowed readers to easily include them.

For each location mention, we applied the GeoNames index and search strategy to generate candidate entries from GeoNames. To select the best candidate from the list, we trained a supervised learning classifier to compare the location mention and the various metadata for each entry from the ontology. This classifier was integrated into Eidos as a Finder and released to Maven Central.



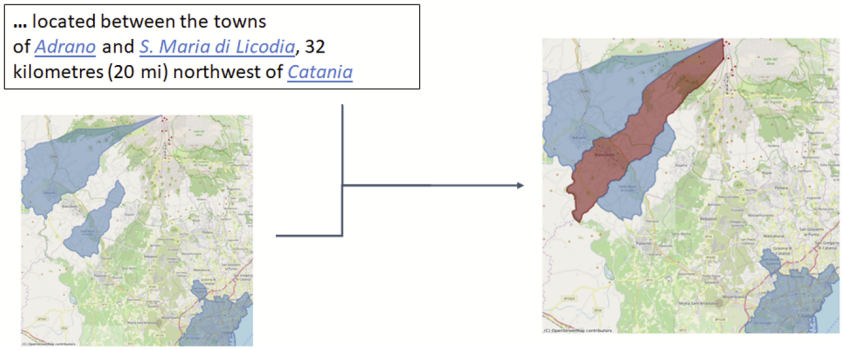
Over the course of the project, we made the following improvements to this architecture:

- Moved from a logistic regression classifier to a transformer-based classifier to to allow more robust matching between mention and entry names.
- Incorporated demonym files from Wikipedia and Stanford CoreNLP into the GeoNames index, yielding about a 10% improvement in performance.
- Introduced a two-stage resolution algorithm where countries, states, counties are resolved first and then used as context to resolve the other locations. This yielded about another 10% improvement in performance.

In our final evaluation, the location identification system achieved 0.500 F1 on the World Modelers evaluation corpus for location identification.

4.5. Annotated corpus of compositional geolocations

To help deal with regions like "surrounding areas in Upper Nile" that did not have entries in the GeoNames ontology, we developed an annotated corpus of geographic locations that were described compositionally in terms of other locations. By combining Wikipedia articles and OpenStreetMap shapefiles, we were able to collect more than 300,000 compositional geographic location phrases and corresponding polygons. The resulting corpus is structured to allow for a task where a model is given a text describing an unnamed location, and is asked to approximate the geometry of that location from the geometries of the geolocations that appear in the description:



We designed a rule-based system to establish the baseline for the task, and integrated it into Eidos. The work was published at COLING 2020 (Laparra and Bethard 2020).

4.6. References

Bethard, Steven. 2013. “A Synchronous Context Free Grammar for Time Normalization.” In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 821–26. Seattle, Washington, USA: Association for Computational Linguistics. https://aclanthology.org/D13-1078.

Bethard, Steven, and Jonathan Parker. 2016. “A Semantically Compositional Annotation Scheme for Time Normalization.” In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*, 3779–86. Portorož, Slovenia: European Language Resources Association (ELRA). https://aclanthology.org/L16-1599.

Laparra, Egoitz, and Steven Bethard. 2020. “A Dataset and Evaluation Framework for Complex Geographical Description Parsing.” In *Proceedings of the 28th International Conference on Computational Linguistics*, 936–48. Barcelona, Spain (Online): International Committee on Computational Linguistics. https://doi.org/10.18653/v1/2020.coling-main.81.

Laparra, Egoitz, Dongfang Xu, and Steven Bethard. 2018. “From Characters to Time Intervals: New Paradigms for Evaluation and Neural Parsing of Time Normalizations.” *Transactions of the Association for Computational Linguistics* 6: 343–56. https://doi.org/10.1162/tacl_a_00025.

Xu, Dongfang, Egoitz Laparra, and Steven Bethard. 2019. “Pre-Trained Contextualized Character Embeddings Lead to Major Improvements in Time Normalization: A Detailed Analysis.” In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, 68–74. Minneapolis, Minnesota: Association for Computational Linguistics. https://doi.org/10.18653/v1/S19-1008.

Yadav, Vikas, Egoitz Laparra, Ti-Tai Wang, Mihai Surdeanu, and Steven Bethard. 2019. “University of Arizona at SemEval-2019 Task 12: Deep-Affix Named Entity Recognition of Geolocation Entities.” In *Proceedings of the 13th International Workshop on Semantic Evaluation*, 1319–23. Minneapolis, Minnesota, USA: Association for Computational Linguistics. https://doi.org/10.18653/v1/S19-2232.

Yadav, Vikas, Rebecca Sharp, and Steven Bethard. 2018. “Deep Affix Features Improve Neural Named Entity Recognizers.” In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, 167–72. New Orleans, Louisiana: Association for Computational Linguistics. https://doi.org/10.18653/v1/S18-2021.

5. Top-down Modeling

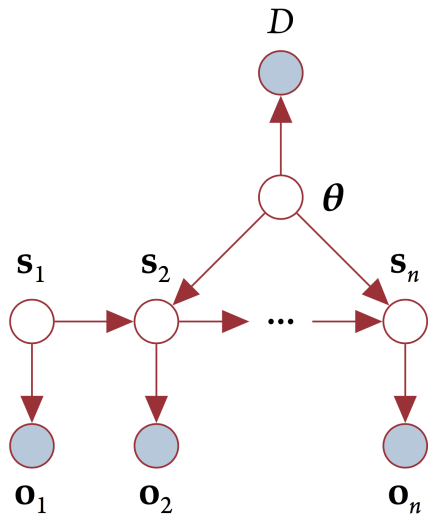
The top-down modeling effort within the UArizona team has developed the Delphi system for assembling executable models from textual evidence and time-series data. For more details, please see https://ml4ai.github.io/delphi/

Modeling approach

We model abstract concepts such as food security and conflict as real-valued latent variables, and indicators corresponding to these concepts as observed variables. An indicator is a measurable quantity that correlates with the intuitive, natural language interpretation of the abstract concept. For example, average dietary energy supply adequacy is one among many indicators for the concept of food security.

We interpret sentences such as “A small increase in conflict causes a large decrease in food security.” as saying something about the functional relationship between two latent variables representing conflict and food security respectively - that is, the sentence is giving us a clue about the shape of $\frac{\partial(food\ security)}{\partial(conflict)}$.

To convert a sentence like the one above to a dynamic Bayes network, we first construct latent states composed of the variables and their partial derivatives with respect to time. In our model, the transition function that tells us how to calculate the latent state at a time step given the latent state at the previous time step is itself a random variable, with a probability distribution constructed from Gaussian kernel density estimators of empirical, crowdsourced data about how humans quantitatively interpret gradable adjectives such as small and large. The reason for this is that there is a great deal of uncertainty in the model itself, since there is no precise model for many abstract concepts such as food security and conflict that we expect to encounter in machine reading. Finally, we use the following emission model - let n_{ij} be the value of the j^{th} indicator for the i^{th} component of a latent state vector **s.** We then define the probability distribution for the value of the indicator as $n_{ij} \sim N(s_i \mu_{ij}, \sigma_{ij})$, where μ_{ij}, σ_{ij} are the mean and standard deviation of the indicator n_{ij}. The structure of the model is shown in the figure below. For more details on the model construction, see here: http://vision.cs.arizona.edu/adarsh/Arizona_Text_to_Model_Procedure.pdf



The figure depicts the model that underlies Delphi - a linear dynamical system with a stochastic transition function. The nodes s_n are latent state vectors, θ is the transition model, and D represents evidence from machine reading.

The capabilities of Delphi are exposed to the users of the CauseMos HMI via a REST API.

User workflows

The Delphi workflow starts with an analyst providing a causal analysis graph (CAG) that informs Delphi how the concepts influence each other.

During the CAG assembly phase, CauseMos, the human-machine interface (HMI) for Delphi, looks up evidence extracted using machine reading on large-scale corpora that relate each edge's source and target concepts. The HMI presents this information to the analyst for scrutiny. Once the analyst has examined the CAG and made any desired modifications, the HMI conveys the CAG (which includes both the metadata about the influence strength attached to each edge and the overall connectivity structure) to Delphi. Delphi uses this information to quantify initial edge strengths and polarities, which serve as a starting point for training a probabilistic forecasting model. While having evidence sentences backing an edge is preferable, it is not required for Delphi to continue training, and it assigns a default edge strength and polarity in such situations.

Notably, Delphi does not require the analyst to manually set the strength or the polarity of any edge. Delphi only consults the information from sentences supplied with the CAG and initializes each edge, and when such information is unavailable, Delphi uses a default. Further, Delphi can work even when the sentences available for an edge do not agree on either the magnitude or the polarity of influence they convey.

Delphi treats each edge weight as a combination of strength and polarity, with the sign of the weight corresponding to the polarity. The weight's magnitude informs Delphi of the size of the change in the target node affected by a change in the source node.

While a CAG is the minimal specification Delphi must receive to build a model, each CAG node should ideally be grounded in indicator time series data to harness the full power of Delphi.

When attaching indicators to CAG nodes during the model creation stage of the HMI, specifying the correct minimum and maximum values for each indicator helps Delphi make better predictions. Delphi does not produce predictions outside the prescribed range for each indicator. Delphi starts training from the starting edge strengths and polarities it gleans from sentences and searches for better edge weights to align the predictions with the training data. In doing so, Delphi does not learn a single model with a single set of parameters; instead, it learns a set of

models that enable Delphi to capture, quantify, and present the uncertainties in the modeling process in the predictions.

When an analyst issues a ‘projection request’, Delphi simulates each model in the set and produces a projected trajectory, resulting in multiple projections per request. The HMI summarizes the projections based on time steps and presents the results to the analyst as the uncertainty of the prediction.

Head nodes

Delphi treats CAG nodes that lack incoming edges, i.e., head nodes differently from other nodes when the analyst specifies a period greater than one and applies a seasonal model. The analyst must supply the accurate seasonality of the time series attached to such CAG nodes as the period for Delphi to model them correctly. If the analyst believes that a CAG node without incoming edges is not seasonal, they could convey it to Delphi by setting its period equal to one.

Editing an edge

After an analyst sees the prediction results of a trained Delphi model, the HMI provides a feature for the analyst to dispute what Delphi has learned as the strength and polarity for an edge. An analyst must supply the weight and the polarity for this purpose.

When Delphi receives such a request, Delphi honors the analyst’s wish and freezes those edges at the weight and polarity assigned by the analyst. Then Delphi goes back to accomplish its mission of generating predictions that resemble the training data by training the model again to learn all the parameters, sans the frozen edges.

Therefore, the analyst should keep in mind that freezing edges initiate a ripple effect where other model parameters could change during the subsequent Delphi training phase accompanied by setting an edge. This could potentially make it seem like (upon visual inspection at least) that freezing edges does not have an effect on the predictions made by Delphi.

Interventions

When an analyst sets the values of indicators at specific time steps, Delphi honors them and outputs those exact values at those time steps. Such changes propagate to downstream nodes, and their predictions will change accordingly.

6. Bottom-up Modeling (UF - DSSAT Team)

6.1. Overview of DSSAT model for World Modeling

The Decision Support System for Agrotechnology Transfer (DSSAT, dssat.net) is a crop modeling system for prediction of crop yields, crop growth dynamics, and soil water and chemical processes. There are many modeling and analytical components included in DSSAT but only a few, described herein, have been exposed to user access in the World Modeling (WM) system.

Figure 1 illustrates a sample DSSAT container as implemented on Dojo for WM. In this example, the DSSAT container includes data and model components for modeling maize in Ethiopia under baseline (current practice) conditions and a few options for perturbing those baseline conditions. When a general modeler makes selections, simulations are performed on Dojo using the DSSAT-pythia modeling framework and the DSSAT-Cropping System model (CSM). The general modeler's selections are encoded into a DSSAT-pythia configuration file and metadata describing the resulting dataset(s).

The main components shown in Figure 1 are:

- Description of the DSSAT container including the user selections for the DSSAT container (i.e., the "tunable knobs") that the general modeler can use to impose scenarios or perturbations to the baseline data
- Description of visualizing DSSAT data in CauseMos
- DSSAT-CSM overview
- DSSAT-CSM inputs including GIS data layers

Model outputs from the simulations are used to generate a dataset which can be used for analysis, mapping, and data exploration. In the WM DSSAT configuration, six output variables are exposed:

1. "crop yield", a measure of the crop productivity per unit area in kg/ha,
2. "crop production", the amount of crop produced which is a product of yield by the area harvested (t),
3. "amount of fertilizer applied", the amount of elemental N in fertilizer applied for crop production (t)
4. "area of crop failure", defined as area in hectares for which the crop yield was less than 200 kg/ha.

Each of these variables is computed for each geographic location, or pixel, in the geospatial simulation and for each of 36 years of historical weather data. Each value is associated with a timestamp year or month and a location (latitude and longitude or administrative unit). More information on the output variables is covered in the section on aggregation.

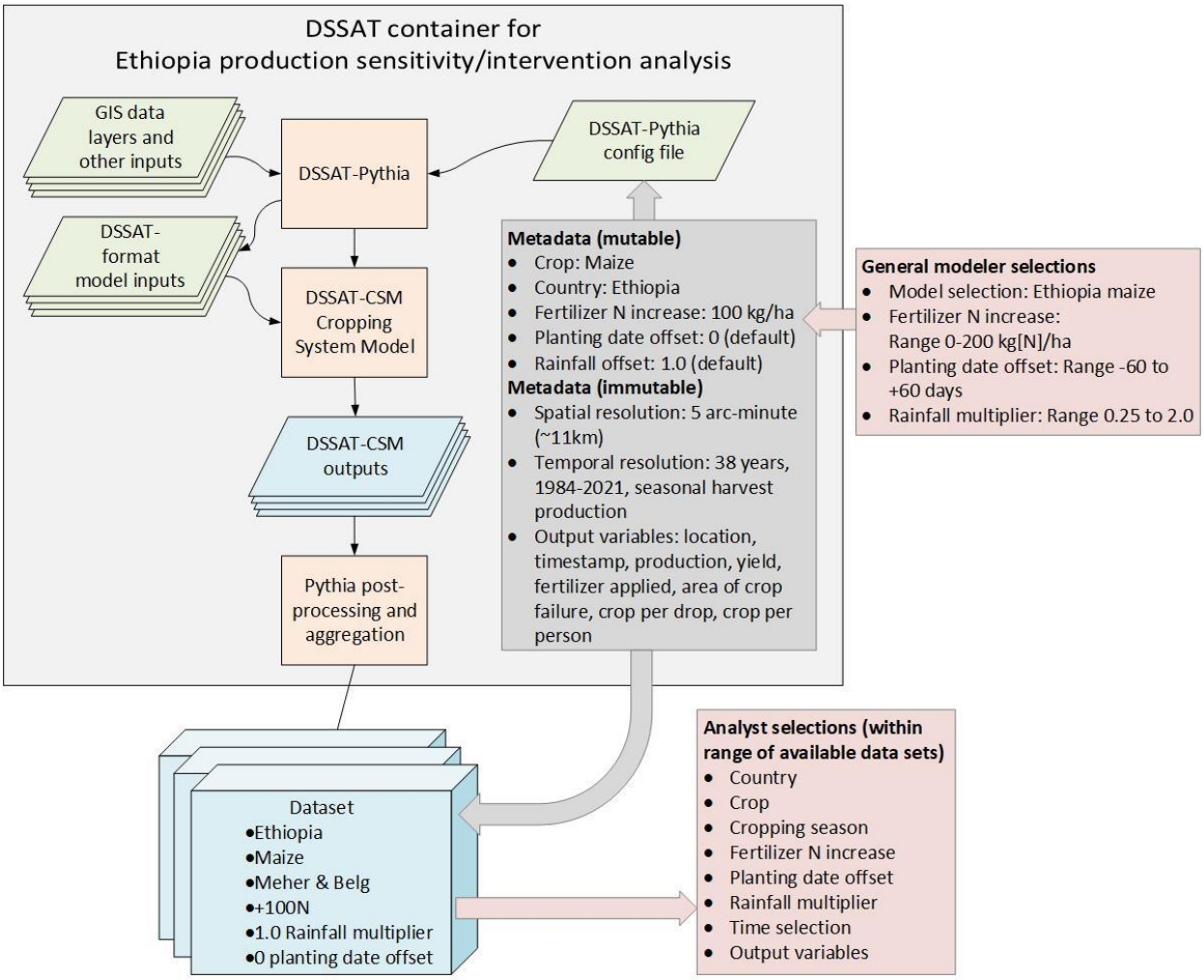


Figure 1. DSSAT container for Ethiopia maize production showing data, processing, and modeling components. A general modeler selects the parameters for a given simulation, which produces a dataset. An analyst accesses the dataset for further analysis and exploration.

DSSAT datasets

DSSAT has additional variables that could be of use to CauseMos users, that have been stored in DSSAT datasets, separately from the DSSAT modeling outputs. These variables require complex aggregation algorithms (e.g., they cannot be simply summed or averaged) and so they were pre-simulated, aggregated, and stored in DSSAT datasets along with some summarizing box plots. These variables include all the variables listed for the model output above plus two additional variables:

1. “crop per drop”, the crop yield divided by the amount of rainfall plus irrigation added to the field in kg/ha/mm. This is a measure of the water use efficiency of a crop.
2. “crop per person” is the crop production in an area divided by the number of people in that area. This is a useful measurement of how well an area might feed it’s people. For urban areas, the number is not as useful a measure, as there are more people with fewer farms. It does not account for imported food or food that is transported from one area to another.

Model types. Three types of DSSAT model are available for use in Dojo and CauseMos. These are:

1. The **DSSAT sensitivity analysis model** includes selections for modifying inputs for fertilizer amounts, planting dates, and rainfall.
2. The DSSAT **climate change model** can be used to modify weather variables to mimic the effects of climate change. The user can select modifications for temperature, rainfall, and atmospheric CO2 concentrations.
3. The DSSAT **yield forecasting model** is used to estimate yield and production for the current cropping season.

Each of these model types is described in separate sections below.

Sensitivity model. Hypothetical scenarios can be imposed by the general modeler to modify baseline (i.e., current farmer practice) conditions. In the WM DSSAT implementation, three input values can be modified to create scenarios:

- Fertilizer inputs can be modified over a range of valid values by selecting a "**Fertilizer N increase**" option, which allows a user to increase the amount of fertilizer applied to fields above the current farmer practice. This is useful for fertilizer subsidy investigations, for example. Values of fertilizer N increase are limited to a few options between zero (the current farmer practice) and +200 kg[N] above current farmer practice.
- Planting dates can be modified by using the "**Planting date offset**" option. Planting dates are simulated in DSSAT using a planting window. On the first day within the planting window for which soil moisture and temperature conditions are within suitable ranges, planting occurs in the model. Baseline planting windows were set by the expert modeler, but these can be shifted by the user to explore early or late planting options. Planting date offset are limited to a few options between -60 (i.e., 60 days prior to current farmer planting dates) to +60 day (i.e., 60 days after current farmer planting dates).
- The "**Rainfall multiplier**" option allows a user to increase or decrease the amount of rainfall during a simulation to mimic effects of drought or higher than normal rainfall. User selection options include a range of values between 0.25 (25% of recorded rainfall amounts) through 2.0 (double the recorded rainfall amounts).

Climate change model. Hypothetical climate scenarios can be imposed by the modeler to study the effects of climate change on crop yield and production and to pinpoint hotspots for climate adaptation recommendations. Three weather input variables can be modified to mimic the effects of climate change: temperature, rainfall, and atmospheric CO_2 levels. These three weather variables affect plant growth processes which often interact in complex ways.

- Temperatures can be modified by selecting the "**Temperature offset**" option. The value selected will be used to increase (or decrease) maximum and minimum daily temperatures in the model. User selections range from "-1" (1 degree C decrease in temperatures) to "+5" (5 degree C increase in temperatures). Plants grow optimally within a range of temperatures which is dependent on the species. Above or below that optimum range, plant growth is stunted. Temperature also affects the rate of progression of plant development.
- As in the sensitivity model, the "**Rainfall multiplier**" option allows a user to increase or decrease the amount of rainfall during a simulation to mimic effects of drought or higher than normal

rainfall. Historical observations of daily rainfall are multiplied by this amount and used as input to the DSSAT model. Optimum soil water conditions improve plant growth, with adverse effects occurring for both too much and too little water. User selections of the rainfall multiplier range from 0.25 to 2.0.

- The "**CO2 value**" option is used to set the value of atmospheric CO_2 for the model simulations. CO_2 generally has a fertilization effect on plant growth, to varying degrees for different species.

Yield forecast model. The yield forecast model is used to predict the crop growth and development in the current or upcoming season. This model relies on recent weather data, ideally updated through yesterday with observed data. The model will use actual weather data for this season up through the last available weather data date. For the remainder of the season, weather data are filled in using recorded data from the last 30 years. This gives 30 possible futures for the upcoming season based on the historical ensemble of weather data. The predicted outcome is based on statistical analysis of the simulated yields. Outcomes are expressed as a distribution of possible production in the coming season.

The only user choice in this mode is the forecast date. These forecast dates have been pre-selected at one-week intervals from March 11, 2022 through May 13, 2022. The user should select the date which has most recently occurred. For example, on if a user is running a prediction on April 12, the most recent available forecast date is April 8. Weather data will be updated just prior to these dates so the user has access to the most recent weather data available for the simulations.

6.1.1. DSSAT models in Dojo

The Dojo DSSAT models were configured by expert modelers who had knowledge of the cropping systems in East Africa and the DSSAT modeling requirements. The number of input variables that are required by a complex model such as DSSAT are too numerous for a WM user to learn on the fly, but the expert modeler has simplified the interfaces to allow only a few selections that can be made by the general modeler. In Figure 1, these are shown in the box to the right, "General modeler selections". These model selections vary depending on the type of model selected.

The general modeler selections are stored as metadata which describe both the simulation configuration and the dataset containing final outputs. These metadata are used to modify the DSSAT-pythia configuration file. Some of the metadata describing the simulation are immutable and cannot be modified by the general modeler, such as the spatial resolution of simulations, which are based on the GIS input data layers.

The DSSAT-pythia config file controls the simulation including parallelization, specifying the location of input data, the number, type, and extent of simulations, and the names and locations of output files. DSSAT-pythia creates the DSSAT-formatted input files from the GIS data layers for the spatial and temporal extents of the simulation. DSSAT-pythia also calls the DSSAT-Cropping System Model (CSM) to simulate for each pixel, year, season, and for the selections made by the general modeler.

A post-processing script takes model outputs and calculates total production and other output variables for each pixel, storing the resulting data in a dataset for further analysis, mapping, and exploration by an analyst.

6.1.2. DSSAT-CSM

The DSST Cropping System Model (CSM) is the main computational element of the Decision Support System for Agrotechnology Transfer (DSSAT, Hoogenboom et al., 2019a and 2019b, Jones et al. 2003). DSSAT-CSM calculates expected growth and development of crops based on equations that describe response to management, environmental conditions, and genetic potential. The equations used in CSM comprise a set of differential equations representing rates of growth or development, and rates of soil water and soil nutrient dynamics. Numerical integration over time, with daily or hourly time steps, allows estimation of growth, development, and water and nutrient levels in the plant and soil.

The 42 crop species currently modeling by DSSAT-CSM deal primarily with annual crops including wheat, rice, maize, and various grain legumes but also include herbaceous perennials such as forage legumes and grasses, and vegetable crops such as tomato and peppers. In addition to crop growth and development, the models simulate water and nutrient dynamics in the soil and crop, so processes such as leaching, organic matter decomposition, and runoff are also considered.

DSSAT-CSM requires input data including daily weather data, soil physical and chemical properties, and detailed crop management, such as timing and details of planting, irrigation, fertilizer application, and residue management. An especially challenging set of inputs are the genotype-specific parameters (GSPs) used to quantify how one cultivar differs from another. GSPs are most often estimated through calibration to measurements from field trials.

DSSAT-CSM has been used for a wide range of applications at different spatial and temporal scales. This includes on-farm and precision management, regional assessments of the impact of climate variability and climate change, gene-based modeling and breeding selection, water use, greenhouse gas emissions, and long-term sustainability through the soil organic carbon and nitrogen balances. Model applications range from real-time decision support for crop management to assessing the potential impact of climate change on global food security. Crop models are also invaluable as heuristic tools that help identify research problems where our current knowledge has limits and further research is needed. The ability of crop models to simulate how different weather years or soil conditions affect crop performance make models especially useful in research involving climatic uncertainty or geospatial variation.

DSSAT has been used by more than 16,500 researchers, educators, consultants, extension agents, growers, and policy and decision makers in over 174 countries worldwide. A more complete overview of DSSAT structure and algorithms is available on dssat.net. Further DSSAT resources can be found in the “Links to DSSAT Resources” below.

6.1.3. DSSAT-CSM model input and output data

Each simulation of DSSAT-CSM requires four types of input data: crop management, weather, soils, and genetics. The following sections describe how each of these are parameterized for the Ethiopia WM Experiment for the four crops provided: maize, wheat, teff, and sorghum.

Although DSSAT-CSM was developed as a field scale model, simulating conditions at a single location, it can be used in a spatial application by gridding the land surface and providing representative model inputs for each pixel on which a crop is grown. For WM, we use a 5 arc-minute resolution (about 10 km) and provide data on soil parameters, daily weather data, and harvested areas as raster data at that resolution or higher. These raster data are described under “GIS data layers” below.

For Ethiopia, two seasons can be simulated for each crop: meher, the main cropping season, and belg, the shorter season. Some areas have only one unimodal season, generally associated with the meher.

These cropping seasons and the associated planting data are discussed further in Section 2.2.1 "Planting information".

Four different management regimens are simulated, representing irrigated, high fertilizer; rainfed, high fertilizer; rainfed, low fertilizer; and rainfed, no fertilizer. Each of these management types is associated with a different harvested area in each pixel of land area.

Management data for baseline conditions, i.e., for current farmer practice, define the crops and cultivars planted, planting dates and densities, fertilizer application rates, irrigation rates, and other management practices. Once the baseline conditions have been fully described, scenarios can be imposed by perturbing these data. For example, the “tunable knob” selecting an amount of increase to fertilizer application rates allows a user to ask “what if” questions, such as “What would be the expected increase in production if we subsidized fertilizers in Ethiopia”?

Each simulation consisting of a crop, cropping season, scenario, management regimen, and pixel is simulated for 36 years of weather data, 1984 through 2019. High spatial variability results from combinations of cropping season, management practices, soils, weather, harvested areas, and genetics which influence production across the country.

6.1.4. DSSAT-pythia post-processing

DSSAT-CSM predicts yield and other variables for each simulation. A post-processor is used to compute production (yield multiplied by harvested area) and to aggregate production for the four management regimens. A simplified set of output data, produced for WM interfaces, contains only the location (latitude and longitude), harvest date, and production. These are stored in datasets grouped by crop, cropping season, and scenario.

6.2. Detailed description of Model inputs and outputs

Details of model inputs are provided for Ethiopia. Other countries data were prepared in a similar manner.

6.2.1. GIS data layers

Weather data: Daily solar radiation and maximum and minimum temperature were obtained from NASA POWER (Zhang et al., 2007, power.larc.nasa.gov). Daily rainfall was collected from CHIRPS (Funk et al., 2015, legacy.chg.ucsb.edu/data/chirps/index.html). The resolution for the NASA POWER data is 0.5 × 0.5 degrees and for the rainfall data is 0.05 × 0.05 degrees. Data were merged at the higher resolution for use as DSSAT model inputs.

Soil data: The soil database corresponds to the Global High-Resolution Soil Profile Database (Han et al., 2015a, b, dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/1PEEY0). Soil organic carbon was considered to be too high within this database and therefore, the soil organic carbon from the Harvest Choice 27 generic soil database (Koo and Dimes, 2010, dataverse.harvard.edu/dataset.xhtml?persistentId=hdl:1902.1/20299) was selected to be used for this project.

Harvested areas: For each pixel, four management regimens are defined. Harvested areas for each of the management types is provided by the SPAM Global spatially disaggregated crop production statistics

data for 2010, version 1.1-version 3. (IFPRI, 2019, https://www.mapspam.info/data/). These define cropland harvested areas for each crop within each geographic pixel at a 5 arc-minute resolution. Harvested areas are provided for four management types: irrigated, high fertilizer input; rainfed, high fertilizer input; rainfed, low fertilizer input; and rainfed, no fertilizer. Field crop management practices are defined separately for the four management types as described below. These harvested areas are used in a post-processor with DSSAT-simulated yields to compute production in each pixel and for each management type. These can then be scaled up to any specified areal boundary, such as country-level or woreda-level production.

Planting zones: For WM simulations, Ethiopia was divided into three planting zones which define ranges of planting dates for each crop and each cropping season. Figure 2 shows the zones which define cropping seasons, planting dates, and cultivars as described in the "Other fixed DSSAT inputs and modeling assumptions" section below. These zones were defined based on rainfall patterns as described in "An atlas of Ethiopian livelihoods" (USAID, 2006).

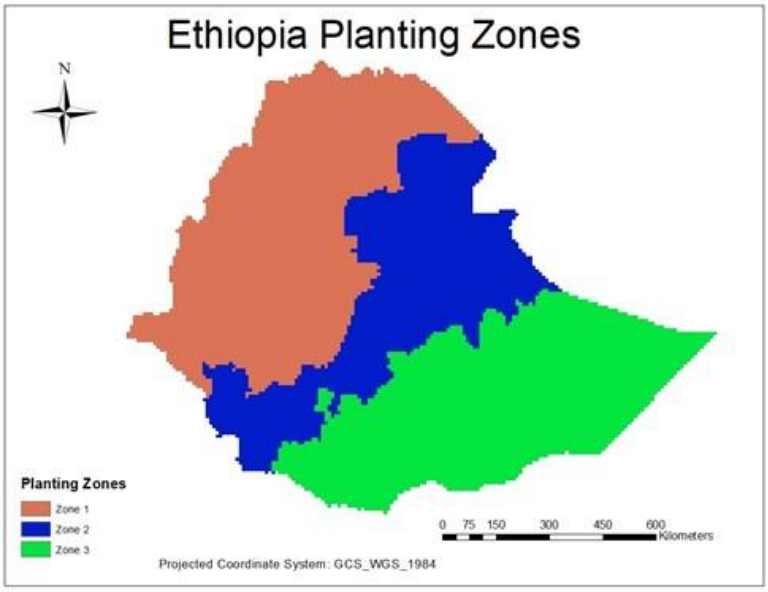


Figure 2. Ethiopia planting zones.

6.2.2. Other fixed DSSAT inputs and modeling assumptions

6.2.2.1 Planting information

Planting dates are defined in Table 2 by crop and season and by planting zone. The dates in Table 2 represent a planting window, or range of possible dates for planting. In the DSSAT-CSM model, crop planting occurs on the first day within this window for which soil moisture and temperature are within acceptable ranges. These planting date windows are based on FAO (2019) crop calendar data, on published papers in the region (various), and on personal communication with agronomists in Ethiopia.

Table 2. Planting dates for each crop, season, and planting zone

Crop - Season	Planting dates for each Zone (Figure 1) and (percent of harvested area allocated to each planting season)		
	Zone 1 North & West	**Zone 2 Central**	**Zone 3 South & East**
Maize, sorghum – Meher	May 1 – Jun 12	Jun 1 – Jul 13	Sep 1 – Oct 13
Maize, sorghum – Belg	--	Feb 1 – Mar 15	Mar 1 – Apr 12
Wheat – Meher	May 1 – June 30	Jun 1 – Jul 31	Sep 1 – Oct 31
Wheat – Belg	--	Feb 1 – Apr 2	Mar 1 – Apr 30
Teff – Meher	May 1 – Jun 12	Jun 1 – Jul 13	Sep 1 – Oct 13
Teff - Belg	--	Feb 1 – Mar 15	Mar 1 – Apr 12

SPAM data layers provide a total harvested area for each crop and each of four management types. The expert modeler made assumptions regarding how these harvested areas are partitioned to Meher and Belg seasons as shown in Table 3. These assumptions were gleaned from literature and statistics and evaluated in aggregate for the country based on CSA reports.

Table 3. Partitioning of harvested area between Meher and Belg seasons

Cropping Season	Zone 1 North & West	Zone 2 Central	Zone 3 South & East
Meher (all crops)	100%	75%	25%
Belg (all crops)	0%	25%	75%

Cultivars selected for these simulations are shown in Table 4 along with the references for each.

Table 4. Crop cultivars used in WM simulations

Crop	Season	Cultivar	Reference
Maize	Meher	BH660	Araya et al. (2015)
Maize	Belg	DSSAT short season	Hoogenboom et al. (2019b)
Sorghum	Meher & Belg	TESHALE	Getachew et al. (2021)

Wheat	Meher & Belg	HAR2501	Araya et al., (2019)
Teff	Meher & Belg	DZ-01-354	Teklu & Tefera (2005), Paff & Asseng (2019)

6.2.2.2 Management regimens

Four management regimens are simulated for each pixel, each with different management inputs. These management regimens are linked to the harvested areas in the SPAM GIS data layers, as described above. Table 3 lists the model input data that are based on the four management regimens for all crops.

Table 5. Management inputs by crop and management regimen

Variable	Management regimen	Maize	Wheat	Teff	Sorghum
Irrigation	Irrigated, high N	Automatic, computed by model			
	Rainfed, high N	None			
	Rainfed, low N	None			
	Rainfed, no N	None			
Fertilizer, kg[N]/ha	Irrigated, high N	100	100	50	100
	Rainfed, high N	100	100	50	100
	Rainfed, low N	10	10	10	10
	Rainfed, no N	0	0	0	0
Applied manure (kg dry matter/ha)	Irrigated, high N	0	0	0	0
	Rainfed, high N	0	0	0	0
	Rainfed, low N	500	500	500	500
	Rainfed, no N	500	500	500	500
Planting density (plants/m2)	All	5	250	900	9
Row spacing (cm)	All	70	16	7	50
Planting depth (cm)	All	5	3	0.9	3

Fertilizer was assumed to be applied in two equal applications on the day of planting and 30 days after planting. The fertilizer type was assumed to be urea, broadcast over the field, then incorporated into the topsoil to a depth of 5 cm.

6.2.3. DSSAT outputs

The standard DSSAT output as implemented in WM includes only a few variables: yield, production,

which is geo-located and timestamped with the harvest date. Production is computed from DSSAT-CSM simulated yield multiplied by the harvested area from SPAM data layers. Total production for each pixel is the sum of production from the four management types. Harvest dates associated with production for each pixel allows a timeline to be constructed over the period of simulation.

There are over 1,200 variables that can be output from DSSAT-CSM simulations. For a full listing of DSSAT output variables, refer to this document. A few that would be most useful for expansion of WM interfaces include: harvested yield, harvested area, planting date, biomass production, plant nitrogen uptake, grain nitrogen (an indicator of protein concentration), soil nitrogen mineralization (a measure of soil nutrient status), precipitation during the growing season, and average temperature during the growing season.

6.3. DSSAT-CSM Calibration and Evaluation

DSSAT-CSM was calibrated for use in East African conditions and evaluated with observed data to instill user confidence in the simulated outputs. Genetic parameters for use in the country-level simulations were obtained by performing detailed calibrations using field crop experimental data. Each cultivar for the crops simulated for WM were selected based on their suitability and representation of farmer practice in Ethiopia. Calibration of these cultivars was performed either by WM expert modelers or were taken from the literature as cited herein.

Evaluation of model outputs is done by aggregating simulated outputs to region or country level and comparing to reported yields and productions at those administrative levels. Comparison over multiple years is used to evaluate the model response to weather variances. Simulated and aggregated outputs for each crop were evaluated with data available from Ethiopian Central Statistics Agency (CSA), the Food and Agriculture Organization (FAO), and the US Department of Agriculture, where data were available.

6.4. Additional Resources

6.4.1 Links to DSSAT resources

DSSAT main website: https://dssat.net/
DSSAT repositories: https://github.com/dssat
DSSAT-CSM repository: https://github.com/dssat/dssat-csm-os
DSSAT-CSM list of output variables definitions:
https://github.com/DSSAT/dssat-csm-os/blob/develop/Data/DATA.CDE
DSSAT documentation: https://github.com/dssat/documentation
DSSAT-pythia documentation: https://pythia-framework.readthedocs.io/en/latest/
DSSAT-pythia-aggregation: https://github.com/DSSAT/supermaas-aggregate-pythia-outputs

6.4.2 References

Araya, A., Hoogenboom, G., Luedeling, E., Hadgu, K. M., Kisekka, I., and Martorano, L. G. (2015). Assessment of maize growth and yield using crop models under present and future climate in southwestern Ethiopia. *Agricultural and Forest Meteorology* 214, 252-265.

Araya, A., Prasad, P. V. V., Gowda, P. H., Afewerk, A., Abadi, B., and Foster, A. J. (2019). Modeling irrigation and nitrogen management of wheat in northern Ethiopia. *Agricultural Water Management* 216, 264-272.

FAO (2019). "GIEWS Country brief Ethiopia." Food and Agriculture Organization of the United Nations.

Funk, C., Peterson, P., Landsfeld, M., Pedreros, D., Verdin, J., Shukla, S., Husak, G., Rowland, J., Harrison, L., Hoell, A., and Michaelsen, J. (2015). The climate hazards infrared precipitation with stations - a new environmental record for monitoring extremes. Scientific data 2, 150066. Earth Engine. https://developers.google.com/earth-engine/datasets/catalog/UCSB-CHG_CHIRPS_DAILY.

Getachew, F., Bayabil, H.K., Hoogenboom, G., Teshome, F.T., Zewdu, E. 2021. Irrigation and Shifting Planting Date as Climate Change Adaptation Strategies for sorghum. Agricultural Water Management. Vol 255, 106988. https://doi.org/10.1016/j.agwat.2021.106988

Han, E., Ines, A., and Koo, J. (2015a). Global high-resolution soil profile database for crop modeling applications. (H. Dataverse, ed.).

Han, E., Ines, A., and Koo, J. (2015b). Global high-resolution soil profile database for crop modeling applications. Working paper. *HarvestChoice/International Food Policy Research Institute (IFPRI)*, 37 pp.

Hoogenboom, G., C.H. Porter, K.J. Boote, V. Shelia, P.W. Wilkens, U. Singh, J.W. White, S. Asseng, J.I. Lizaso, L.P. Moreno, W. Pavan, R. Ogoshi, L.A. Hunt, G.Y. Tsuji, and J.W. Jones. 2019a. The DSSAT crop modeling ecosystem. In: p.173-216 [K.J. Boote, editor] Advances in Crop Modeling for a Sustainable Agriculture. Burleigh Dodds Science Publishing, Cambridge, United Kingdom (http://dx.doi.org/10.19103/AS.2019.0061.10)

Hoogenboom, G., C.H. Porter, V. Shelia, K.J. Boote, U. Singh, J.W. White, L.A. Hunt, R. Ogoshi, J.I. Lizaso, J. Koo, S. Asseng, A. Singels, L.P. Moreno, and J.W. Jones. 2019b. Decision Support System for Agrotechnology Transfer (DSSAT) Version 4.7.5 (https://DSSAT.net). DSSAT Foundation, Gainesville, Florida, USA.

IFPRI (2019). Global spatially-disaggregated crop production statistics data for 2010 Version 1.1. (H. Dataverse, ed.).

Jones, J.W., G. Hoogenboom, C.H. Porter, K.J. Boote, W.D. Batchelor, L.A. Hunt, P.W. Wilkens, U. Singh, A.J. Gijsman, and J.T. Ritchie. 2003. DSSAT Cropping System Model. European Journal of Agronomy 18:235-265.

Koo, J., and Dimes, J. (2010). HC27: Generic/Prototypical Soil Profiles. International Food Policy Research Institute, Washington, DC., and University of Minnesota, St. Paul, MN. Available online at http://harvestchoice.org/node/2239.

Paff, K., and Asseng, S. (2019). Comparing the effects of growing conditions on simulated Ethiopian tef and wheat yields. *Agricultural and Forest Meteorology* 266, 208-220.

Teklu, Y., and Tefera, H. (2005). Genetic improvement in grain yield potential and associated agronomic traits of tef (*Eragrostis tef*). *Euphytica* 141, 247-254.

USAID (2006). "An atlas of Ethiopian livelihoods. The livelihoods integration unit." International Food Policy Research Institute, Washington DC. / Central Statistical Agency / Ethiopian Development Research Institute, Addis Ababa, Ethiopia.
http://foodeconomy.com/wp-content/uploads/2016/02/Atlas-Final-Web-Version-6_14.pdf

Zhang, T., Chandler, W. S., Hoell, J. M., Westberg, D., Whitlock, C. H., and Stackhouse Jr, P. W. (2007). A Global perspective on renewable energy resources: NASA's Prediction of Worldwide Energy Resources (POWER) project. In "Proceedings of ISES World Congress 2007" (G. D.Y. and Z. Y., eds.), Vol. I-V, pp. 2636-2640, Berlin, Heidelberg, Germany.