> FHI TECHNICAL REPORT <

QNRs
Toward Language for Intelligent Machines

K. Eric Drexler

Senior Research Fellow
eric.drexler@oxfordmartin.ox.ac.uk

Technical Report #2021-3

Cite as:
Drexler, K.E. (2021): "QNRs: Toward Language for Intelligent Machines", Technical Report #2021-3, Future of Humanity Institute, University of Oxford

Abstract

Impoverished syntax and nondifferentiable vocabularies make natural language a poor medium for neural representation learning and applications. Learned, quasilinguistic neural representations (QNRs) can upgrade words to embeddings and syntax to graphs to provide a more expressive and computationally tractable medium. Graph-structured, embedding-based quasilinguistic representations can support formal and informal reasoning, human and inter-agent communication, and the development of scalable quasilinguistic corpora with characteristics of both literatures and associative memory.

To achieve human-like intellectual competence, machines must be fully literate, able not only to read and learn, but to write things worth retaining as contributions to collective knowledge. In support of this goal, QNR-based systems could translate and process natural language corpora to support the aggregation, refinement, integration, extension, and application of knowledge at scale. Incremental development of QNR-based models can build on current methods in neural machine learning, and as systems mature, could potentially complement or replace today's opaque, error-prone "foundation models" with systems that are more capable, interpretable, and epistemically reliable. Potential applications and implications are broad.

To facilitate skimming, brief summaries of the main sections of this document are collected in Section 2.5.

Contents

1 Introduction

This section presents a brief summary and outline of core concepts, including boundaries (what is *not* proposed) and some terminology. Descriptions of the main sections are collected in Section 2.5.

1.1 A brief summary

Natural language (NL) is a powerful medium for expressing human knowledge, preferences, intentions, and more, yet NL words and syntax appear impoverished when compared to the representation mechanisms (vector embeddings, directed graphs) available in modern neural ML. Taking NL as a point of departure, we can seek to develop representation systems that are strictly more expressive than natural language. The approach proposed here combines graphs and embeddings to support quasilinguistic neural representations (QNRs) shaped by architectural inductive biases and learned through multitask training. Graphs can strongly generalize NL syntactic structures, while lexical-level embeddings can strongly generalize NL vocabularies. QNR frameworks can syntactically embed and wrap non-linguistic objects (images, data sets, *etc.*) and formal symbolic representations (source code, mathematical proofs, *etc.*). Through access to external repositories (Figure 1.1), inference systems can draw on corpora with content that spans scales that range from phrases and documents to scientific literatures and beyond.

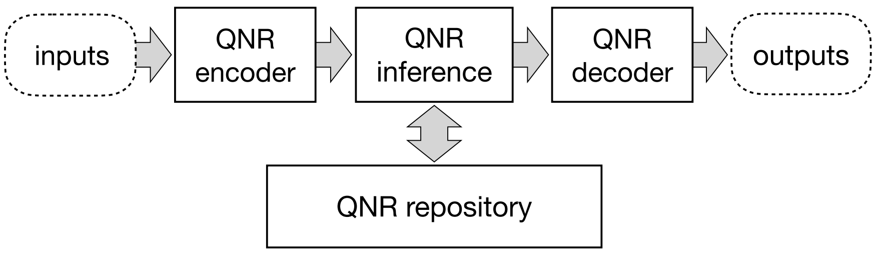

Figure 1.1: *Information flows in generic QNR systems supported by access to a repository of QNR content. Inputs and outputs may be multimodal.*

Embeddings can abstract QNR content to enable semantic associative memory at scale. Neural networks, potentially exploiting (soft) lattice operations, can process retrieved QNR content to recognize analogies, complete patterns, merge compatible descriptions, identify clashes, answer questions, and integrate information from both task inputs and repositories.

"NL$^+$" refers to aspirational QNR systems that outperform natural language as a medium for semantic expression and processing. The NL$^+$ vision aligns with and extends current research directions in NLP, and NL$^+$ implementations could build on current neural architectures and training methods.

Potential applications are diverse, ranging from familiar NL-to-NL functionality (interactive search, question answering, writing, translating) to novel forms of representation and reasoning in science, engineering, software development, and mathematics. Potential advantages in scalability, interpretability, cost, and epistemic quality position QNR-based systems to complement or displace opaque foundation models (Bommasani *et al.* 2021) at the frontiers of machine learning.

To facilitate skimming, brief summaries of the main sections are collected in Section 2.5. Readers who prefer to start in the middle may wish to skip ahead to Section 8: Quasilinguistic Neural Representations.

1.2 Some (Over)simplified Descriptions

An oversimplified problem framing:

human intelligence : natural language :: machine intelligence : _______?

An oversimplified approach: Use architectural inductive bias and representation learning in neural ML systems to upgrade language by replacing word sequences with explicit parse trees and words with embedding vectors. This is an *over*simplification because it (wrongly) suggests a close, fine-grained correspondence between natural languages and QNRs.

A less oversimplified description: Use architectural inductive bias and representation learning to develop models that generate and process directed graphs (that *strongly generalize* NL syntax) labeled with vector embeddings (that *strongly generalize* both NL words and phrases), thereby subsuming and extending both the syntactic structures and lexical-level components of natural languages. The resulting representation systems can surpass natural languages in expressive capacity, compositionality, and computational tractability.

Further objectives and approaches: Learn to embed lexical-level vector representations in *structured* semantic spaces. Use inductive biases and multitask learning to associate meanings with semantic-space *regions* (rather than points), and exploit approximate lattice operations (soft unification and anti-unification) as mechanisms for knowledge integration, refinement, and generalization. Translate broad knowledge (*e.g.*, from natural language corpora) into large QNR corpora and employ scalable algorithms to access and apply this knowledge to a wide range of tasks. Enable neural ML systems to write and read QNR content to enable learning that is both efficient and interpretable.

1.3 What is *Not* Proposed

Some contrasting negative samples from the space of related concepts can help readers refine their internal representations of the present proposal:

> ***Not*** *a formal language.* Formal languages supplement natural languages, but have never subsumed their expressive capacity; frameworks proposed here can embed but are not constrained by formal representations.
>
> ***Not*** *a constructed language.* Constructed languages[1] have typically sought clarity and comprehensibility, yet sacrificed expressive capacity; frameworks proposed here seek to expand expressive capacity, yet as a consequence, sacrifice full human comprehensibility.
>
> ***Not*** *a system of hand-crafted representations.* Products of neural representation learning typically outperform hand-crafted representations; accordingly, frameworks proposed here rely, not on hand-crafted representations, but on representation learning shaped by architectural bias and training tasks.[2]
>
> ***Not*** *a radical departure from current neural ML.* Frameworks proposed here are informed by recent developments in neural ML and suggest directions that are aligned with current research.

1.4 Some Concepts and Terms

- "NL" refers to natural language in a generic sense. The representational capacity of NL (in this sense) can be thought of as a sum of the representational capacities of human languages.
- Representations will be vector-labeled graphs (VLGs); potential arc labels (indicating types, *etc.*) are not explicitly discussed.

1. Lingua generalis, Esperanto, Loglan, *etc.*

2. This document often describes *illustrative forms* of representation and functionality, or describes how neural computation *could potentially* implement those forms and functions, but always with the implicit proviso that learned neural representations and mechanisms are apt to be surprising.

- Quasilinguistic neural representations (QNRs, implemented as VLGs) are compositional and language-like: graphs provide upgraded syntactic structure, while embeddings provide upgraded lexical components.[1]
- "NL^{+}" refers to proposed[2] QNR-based products of neural representation learning that would subsume and extend the representational capacity of natural languages.[3]
- The term "lattice" and the lattice operations of "meet" (here, "unification") and "join" (here, "anti-unification", sometimes termed "generalization") have their usual mathematical meanings; in the present context, however, lattices and lattice operations will typically be approximate, or "soft" (Appendix A1).

2 Motivation and Overview

Several perspectives converge to suggest that high-level machine intelligence will require literacy that is best developed in a machine-native medium that is more expressive than natural language. This section concludes with an overview of the sections that follow.

Because language and machine learning are broad topics intertwined with each other and with a host of disciplines and application fields, it is difficult to neatly disentangle the various "motivations and perspectives" promised by the section title. The discussion that follows (perhaps unavoidably) contains sections with overlapping conceptual content.

2.1 Why Look Beyond Natural Language?

Why seek a language-like representational medium that is more expressive and computationally tractable than natural language? The question almost answers itself. But is such a medium possible, what would it be like, how might it be developed and applied? More generally, how might we complete the analogy mentioned above,

1. Formal language-like systems (programming languages, mathematical notations, *etc.*) are sometimes called "quasilinguistic"; here, the term is extended to include less formal systems.

2. Here, to "propose" means to suggest a potential future objective or development; in the ML literature, by contrast, what is "proposed" is often already demonstrated.

3. Superscripting "+" improves esthetics in hyphenated forms; using the U+207A character code improves typographic stability.

human intelligence : natural language :: machine intelligence : _________?

It seems unlikely that the best answer is "natural language" (again) or "unstructured vector embeddings".

Human intelligence and human societies rely on language as a primary medium for communicating and accumulating knowledge, for coordinating activities, and to some substantial extent, for supporting individual cognition. Intellectually competent humans are literate: They can read and can write content worth reading. High-level machine intelligence will surely be able to do the same and have use for that ability. Current AI research is making strong progress in reading and writing natural language as an interface to the human world, yet makes little use of language(-like) representations for communicating and accumulating knowledge within and between machines.

The world's accessible information constitutes a vast, multimodal corpus in which natural language serves as both content and connective tissue. General, high-level intelligent systems must be able to use and extend this information, and it is natural to seek a medium for representing knowledge, both translated and new, that is well-adapted and in some sense native to neural machine intelligence.

What might a machine-adapted language be like? It would be strange to find that the best languages for neural ML systems lack basic structural features of human language—in particular, syntax and word-like units—yet perhaps equally strange to find that machines able to share expressive vector embeddings will instead employ sequences of tokens that represent mouth noises.

The present document proposes a framework for quasilinguistic neural representations (QNRs) that—by construction—could match and exceed the representational capacity of natural language. Both the potential value and general requirements for such systems seem clear enough to motivate and orient further investigation.

2.2 Some Motivating Facts and Hypotheses

The motivation for pursuing QNR approaches that are anchored in NL can be grounded both in uncontroversial facts and in contrasting plausible and implausible hypotheses.

Key motivating facts

1) Natural language is a key element of human cognition and communication.

2) Natural language provides expressive capacity of unique breadth and flexibility.
3) Structured neural representations can be both richer and more ML-compatible[1] than sequences of words.

Corresponding (and plausible) motivating hypotheses

+1) Quasilinguistic neural representations of some sort will be key elements of human-level machine cognition and communication, *and:*
+2) The abstract features of natural language (syntactic and lexical constructs) can inform the development of QNRs that subsume and extend syntax and words with graphs and embeddings, *and:*
+3) QNRs informed by natural language constructs can be more expressive and computationally tractable than languages that translate or imitate NL-like sequences of word-like tokens.

Corresponding (but implausible) demotivating hypotheses The plausibility of the above hypotheses is supported by the implausibility of contrary hypotheses:

–1) That language-like representations will be of little use in human-level machine cognition and communication, *or:*
–2) That language-like syntactic and lexical structures can be no better than flat sequences of vector representations,[2] *or:*
–3) That embeddings in combination with language-like syntactic structures can be no more expressive than sequences of word-like tokens.

2.3 General Approach and Goals

In brief, the present line of inquiry suggests a framework that would, as already outlined in part:

1. *E.g.*, they can be differentiable

2. Note that representations of theoretically equivalent expressive capacity need not be equivalent in, for example, computational tractability, compositionality, compactness, scalability, or inductive bias.

- Replace and generalize discrete, non-differentiable NL words and phrases with semantically rich, differentiable embeddings.[1]
- Replace and generalize NL syntax with general graphs (which also have differentiable representations).
- Complement flat neural representations with syntactic structure.
- Move linguistic content closer to (quasi)cognitive representations.

This strategy starts with NL as a point of departure, retaining generality by subsuming and extending NL piecemeal, at the level of understandable elements. The alternative—to attempt to capture the whole of NL functionality in a more formal, theory-based framework—would risk the loss of functionality that we do not fully understand.

Beyond these basic features, QNR frameworks can be extended to:

- Exploit abstractive embeddings of fine-grained content (Section 8.3.4).
- Exploit abstractive embeddings of large-scale contexts (Section 8.3.5).
- Support semantic search at scale (Section 9.1.2).
- Support semantic normalization, alignment, refinement, and integration (Section 8.4).
- Subsume or embed formal and non-linguistic representations (Section 9.2).

What do we want from scalable high-end QNR/NL$^+$ systems?

- To translate (and refine) large NL corpora into more tractable forms[2]
- To combine knowledge from multiple sources, making use of recognizable concordance, clashes, and gaps
- To provide comprehensive, dynamic, beyond-encyclopedic knowledge for use by machines and humans
- To support the growth of knowledge through machine-aided reasoning

The latter goals are worth emphasizing: A key motivation for pursuing NL$^+$ capabilities is to enable systems to learn from, apply, and extend content that ranges from informal, commonsense knowledge to mathematics and scientific

1. One direction in which language-like representations might diverge from the picture painted here is in the semantic level of embeddings: As discussed in Section 5.3, embeddings can, through representation discovery, subsume the function of syntactic units above the lexical level (*e.g.*, relatively complex phrases and relatively simple sentences). The partial interchangeability of graph and vector representations (Section 7.3) blurs the potential significance of such a shift, however.

2. While also translating among a likely multiplicity of NL$^+$ dialects and overlapping task-oriented sublanguages.

literatures. While NL$^+$ representations have potentially important roles in NL-to-NL processing (translation, *etc.*), this is almost incidental. The primary aim is to represent, not NL, but what NL itself represents, and to do so better and with broader scope.

Current language-related machine representations do not provide full NL (much less NL$^+$) functionality: They range from opaque language models to explicit knowledge graphs and formal languages, but despite their strengths, none can match (much less exceed) human language in power and generality. Systems like these should be seen as complements—not alternatives—to QNR frameworks.[1]

2.4 Four Perspectives That Help Situate the NL$^+$ Concept

Reinforcing the points above, four external perspectives may help to situate the NL$^+$ concept with respect to related research topics:

- *The power and limitations of natural language* set a high bar to clear while suggesting directions for developing more powerful systems.
- *The power and limitations of symbolic systems*[2] suggest a need for complementary, less formal representation systems.
- *The power and limitations of flat neural representations* suggest the potential advantages of systems that combine the expressive power of dense vector representations with the compositional structure of NL.
- *The power and limitations of current NLP tools*[3] suggest that current neural ML techniques can both support and benefit from QNR-based mechanisms with NL$^+$ applications.

2.5 Section Overviews

The topics addressed in this document are broad, many-faceted, and have a large surface area in contact with other disciplines. The topics are difficult to disentangle, but the following overviews provide a sketch of the organization and content of the document.[4]

1. For example, opaque Transformer-like models may be useful in QNR applications: In general, quasicognitive processing is complementary to quasilinguistic representation.

2. Logic, mathematics, programming languages, knowledge representation languages, attempted formalizations of natural language, *etc.*

3. Including systems that exploit pretrained language models.

4. Readers who don't skim will encounter redundancy provided for those who do.

Section 1: Introduction This section presents a brief summary and outline of core concepts, including boundaries (what is *not* proposed) and some terminology.

Section 2: Motivation and Overview. Several perspectives converge to suggest that high-level machine intelligence will require literacy that is best developed in a medium more expressive than natural language.

Section 3: Notes on Related Work. Current developments in neural ML provide architectures and training methods that can support QNR-oriented research and development. Prospective advances are linked to work in symbolic and neurosymbolic computation, and to broad trends in deep learning and natural language processing.

Section 4: Language, Cognition, and Neural Representations. Using NL as a motivation and point of departure for NL$^+$ motivates a review of its roles in communication, cognition, and the growth of human knowledge. Prospects for improving compositionality through QNR representations are key considerations.

Section 5: Expressive Constructs in NL and NL$^+$. NL$^+$ must subsume the functionality of NL constructs identified by linguists, and the shortcomings of those constructs suggest substantial scope for surpassing NL's expressive capacity.

Section 6: Desiderata and Directions for NL$^+$. Prospects for improving expressiveness in NL$^+$ representations include mechanisms both like and beyond those found in natural languages. Research directions aimed at realizing these prospects are well-aligned with current directions in neural ML.

Section 7: Vector-Labeled Graph Representations. In conjunction with today's deep learning toolkit, vector-labeled graph representations provide powerful, differentiable mechanisms for implementing systems that represent and process structured semantic information.

Section 8: Quasilinguistic Neural Representations. Applications of vector-labeled graphs can generalize NL syntax and upgrade NL words to implement quasilinguistic neural representations that parallel and surpass the expressive capacity of natural language at multiple levels and scales.

Section 9: Scaling, Refining, and Extending QNR Corpora. Scalable QNR systems with NL$^+$-level expressive capacity could be used to represent, refine, and integrate both linguistic and non-linguistic content, enabling systems to compile and apply knowledge at internet scale.

Section 10: Architectures and Training. Extensions of current neural ML methods can leverage architectural inductive bias and multitask learning to support the training of quasilinguistic neural systems with NL$^+$-level expressive capacity.

Section 11: Potential Application Areas. Potential applications of QNR/NL$^+$ functionality include and extend applications of natural language. They include human-oriented NLP tasks (translation, question answering, semantic search), but also inter-agent communication and the integration of formal and informal representations to support science, mathematics, automatic programming, and AutoML.

Section 12: Aspects of Broader Impact. The breadth of potential applications of QNR-based systems makes it difficult to foresee (much less summarize) their potential impacts. Leading considerations include the potential use and abuse of linguistic capabilities, of agent capabilities, and of knowledge in general. Systems based on QNR representations promise to be relatively transparent and subject to correction.

Section 13: Conclusions. Current neural ML capabilities can support the development of systems based on quasilinguistic neural representations, a line of research that promises to advance a range of research goals and applications in NLP and beyond.

2.6 Appendix Overviews

Several topics have been separated and placed in appendices. Of these, only the first focuses on topics that can be considered foundational.

Appendix A1: Unification and Generalization on Soft Semantic Lattices. QNR representations can support operations that combine, contrast, and generalize information. These operations—soft approximations of unification and anti-unification—can be used to implement continuous relaxations of powerful mechanisms for logical inference.

Appendix A2: Tense, Aspect, Modality, Case, and Function Words. Tables of examples illustrate expressive constructs of natural languages that do not reduce to nouns, verbs, and adjectives.

Appendix A3: From NL Constructs to NL$^+$. Condensing, regularizing, and extending the scope of semantic representations can improve expressive capacity and compositionality, and can support theoretically grounded methods for comparing and combining semantic information.

Appendix A4: Compositional Lexical Units. Embeddings with explicit compositional structure may offer advantages in efficient learning and generalization.

Appendix A5: Compact QNR Encodings. String representations of QNRs, in conjunction with discretized vector spaces and graph-construction operators, can provide compact and efficient QNR encodings.

3 Notes on Related Work

Current developments in neural ML provide architectures and training methods that can support QNR-oriented research and development. Prospective advances are linked to work in symbolic and neurosymbolic computation, and to broad trends in deep learning and natural language processing.

The prospects explored in the present document include NLP-oriented QNR systems, which is to say, systems that read, process, and produce content within QNR domains while supporting NL inputs and outputs at external interfaces. The discussion focuses on broad, long-term goals and associated software infrastructure.

Topics considered at this level are too abstract to correspond closely to particular neural ML implementations, precluding fine-grained comparisons. Accordingly, this section discusses connections to current work (useful tools, competing and complementary approaches), but only in outline; more extensive discussions of related work can be found in cited papers.

3.1 Potentially Useful Models and Tools

Potentially useful models and tools for quasilinguistic processing (QLP) are coextensive with broad areas of neural ML (in particular, neural NLP), and a range of applicable tools (architectures, training data, training tasks, computational resources. . .) can be found in current practice.

3.1.1 Vector-Oriented Representations and Algorithms

Flat neural representations—sets and sequences of one or more embedding vectors—are ubiquitous in modern neural ML and play key roles as components of proposed QNR architectures. The most closely related work is in natural language processing.

In neural NLP, we find vector representations of words at input and (often) output interfaces;[1] some systems produce embeddings of higher-level entities such as sentences and documents.[2] Semantic structure in vector spaces emerges spontaneously in word embeddings.[3] End-to-end training can produce compatible vector representations of images and text for tasks that include image captioning and visual question answering.[4]

Extensions of current NLP representations, architectures, and training methods are natural candidates for analogous roles in QLP. Transformer architectures—successful in tasks as diverse as translation, question answering, theorem proving, object recognition, and graph-based inference[5]—appear to have sufficient generality to support many (perhaps most) aspects of quasilinguistic processing.

Transformer architectures have been extended to read external memories, including stores of NL text[6] and vector embeddings.[7] QLP systems could po-

1. Rezaeinia, Ghodsi, and Rahmani (2017) and Devlin *et al.* (2019)

2. Adi *et al.* (2017), Ganguly and Pudi (2017), and Conneau *et al.* (2018)

3. Mikolov, Yih, and Zweig (2013), S. Liu *et al.* (2018), and Ethayarajh, Duvenaud, and Hirst (2019). Relative to words in natural languages, embeddings can improve correspondence between representational and semantic distance, a long-standing goal for improving linguistic systems (Wilkins 1668).

4. Yu *et al.* (2017) and Hossain *et al.* (2019)

5. Vaswani *et al.* (2017), Devlin *et al.* (2019), Koncel-Kedziorski *et al.* (2019), Brown *et al.* (2020), Polu and Sutskever (2020), and Carion *et al.* (2020)

6. *E.g.*, 128-token text pieces (Verga *et al.* 2020), or more general multimodal information (Fan *et al.* 2021).

7. Khandelwal *et al.* (2020) and Yogatama, Masson d'Autume, and Kong (2021). Models of this kind fall within the broad class of memory-augmented neural networks (Santoro *et al.* 2016).

tentially be based on broadly similar architectures in which inference systems write and read, not flat embeddings, but QNR content.

3.1.2 Graph Representations and GNNs

Graph structures complement vector representations in proposed QNRs, and applications of graph representations have spurred extensive work in neural ML.

Iterative, node-to-node message-passing systems—graph neural networks[1] (GNNs)—are deep, convolutional architectures that have been successful in tasks that range from scene understanding to quantum chemistry and neurosymbolic computing;[2] their functionality may be well-suited to semantic processing on QNRs. Graph-oriented models, often GNNs, are widespread in neural knowledge-representation systems.[3] Similar graphs can be aligned for comparison and processing.[4] Although classic GNNs operate on fixed graphs, both differentiable representations and reinforcement learning have been used to implement generative models in the discrete graph-structure domain[5] (graphs can, for example, be mapped to and from continuous embeddings[6]). With suitable positional encodings, Transformers can operate not only on sequences, but on trees or general graphs.[7] The rich tool set provided by current graph-oriented neural models seems sufficient to support the development of powerful QNR-based applications.

3.1.3 Computational Infrastructure

Broad applications of NL$^+$ call for scaling to large corpora, first training on large NL corpora, then writing and applying QNR corpora that may be larger still. Rough analogies between NLP and QLP tasks suggest that computational costs in both training and applying large-scale systems can be in line with the costs of currently practical systems for language modeling and translation

1. Recently reviewed in J. Zhou *et al.* (2020) and Wu *et al.* (2021).

2. R. Li *et al.* (2017), Gilmer *et al.* (2017), Lamb *et al.* (2020), and Addanki *et al.* (2021). Labeled scene graphs, in particular, exemplify learnable semantic relationships among objects in which both objects and their relationships can best be represented by embeddings (see Figure 8.1).

3. Wen Zhang *et al.* (2019) and Ji *et al.* (2021)

4. Heimann *et al.* (2018), Cao *et al.* (2019), and Fey *et al.* (2020)

5. Yun *et al.* (2019), J. Zhou *et al.* (2020), Kazi *et al.* (2020), and Wu *et al.* (2021)

6. Cai, Zheng, and Chang (2018) and Pan *et al.* (2018)

7. Shiv and Quirk (2019) and Chen, Barzilay, and Jaakkola (2019)

(Section 9.1); in particular, algorithms for efficient embedding-based semantic search at scale—a key enabler for exploiting large corpora—have been demonstrated in commercial applications.[1] Accordingly, current computational infrastructure seems adequate for development, training, and potential large-scale deployment of NL$^+$ applications. Reductions in computational cost and improvements in algorithmic efficiency continue (Hernandez and Brown 2020).

3.2 Symbolic, Neural, and Neurosymbolic AI

Classic symbolic and neural approaches to AI provide further context for the proposed line of development, which has connections to combined, neurosymbolic approaches.

3.2.1 Symbolic AI

The early decades of AI centered on symbolic models that have little direct relevance to current neural approaches. Symbolic systems had striking successes,[2] yet produced unimpressive results in learning and perceptual tasks like vision. In NLP, symbolic AI faced persistent difficulties stemming from the interplay of word meanings, syntax, and semantic context, while in a key application—machine translation—statistical methods outperformed classic symbolic AI.

The quasilinguistic approach suggested here differs from symbolic AI in two quite general ways:

1. Proposed QNRs and QLP computation are intended to support—not directly implement—mechanisms for inference and control.
2. Proposed QNRs and QLP computation are saturated with neural representations and learning mechanisms.

What symbolic AI *does* have in common with proposed QLP is the use of graph-structured representations (in symbolic AI, typically syntax trees) that are associated with distinct lexical-level components.

1. J. Wang *et al.* (2018) and Johnson, Douze, and Jégou (2019)

2. *Perceptions* of success were (notoriously) blunted by reclassification of research results ("If it works, it isn't AI"). Highly successful automation of symbolic mathematics, for example, emerged from what had initially been considered AI research (Martin and Fateman 1971; Moses 2012).

3.2.2 Neural ML

In recent years deep learning and neural ML have advanced rapidly in both scope and performance, successfully addressing an astounding range of problems. Because the range of potential neural architectures and tasks is open ended, it would be unwise to draw a line around deep learning and propose limits to its capabilities. The present proposals are within, not beyond, the scope of modern neural ML.

That said, one can point to persistent difficulties with the most common neural ML approaches, which is to say, models that employ flat neural representations (vectors, sets of vectors, sequences of vectors) that often scale poorly, lack clear compositionality, and resist interpretation.

3.2.3 Neurosymbolic AI

Developments in neurosymbolic AI are advancing at the intersection between symbolic and neural ML, with approaches that include the adaptation of symbolic algorithms to richer, embedding-based representations.[1] This body of work has multiple points of contact with proposed QNR approaches, but its diversity resists summarization. Appendix A1 explores connections with constraint logic programming and related reasoning mechanisms based on neurosymbolic representations.

It is important to distinguish among approaches that can be called "neurosymbolic", yet differ fundamentally. Geoffrey Hinton has remarked that:

> *Some critics of deep learning argue that neural nets cannot deal with compositional hierarchies and that there needs to be a "neurosymbolic" interface which allows neural network front- and back-ends to hand over the higher-level reasoning to a more symbolic system. I believe that our primary mode of reasoning is by using analogies which are made possible by the similarities between learned high-dimensional vectors... (Hinton 2021)*

The present proposal differs from those that Hinton criticizes: While both QNRs and conventional symbolic systems employ explicit syntactic structures, compositional hierarchies, and word-like units, QNRs employ high-dimensional vectors, not conventional symbol-tokens, in part for the reason

1. *E.g.*, see Rocktäschel and Riedel (2017), Minervini *et al.* (2020), Garcez and Lamb (2020), and Arabshahi *et al.* (2021).

Hinton cites. Although higher-level reasoning seems likely to have an algorithmic character, employing conditional branches and dispatch of values to functions,[1] there is good reason to expect that those conditionals and functions will operate on neural representations through neural mechanisms. To structure the objects and operations of reasoning need not impoverish their content.

3.3 Foundation models

The term "foundation model" has been has been introduced (Bommasani *et al.* 2021) to describe systems that are "trained on broad data at scale and can be adapted (*e.g.*, fine-tuned) to a wide range of downstream tasks". Today's leading foundation models (*e.g.*, BERT and GPT-3[2]) are pretrained on extensive corpora of NL next, while others (*e.g.*, CLIP[3]) are multimodal; all are based on Transformers.

Despite their extraordinary range of applications, current foundation models have suffered from opaque representations, opaque inference mechanisms, costly scaling,[4] poor interpretability, and low epistemic quality, with consequences reviewed and explored in depth by Bommasani *et al.* (2021).

QNR-oriented architectures could potentially alleviate each of these difficulties by complementing or displacing models based on stand-alone Transformers. Rather than representing knowledge in unstructured, multi-billion-parameter models,[5] architectures that represent knowledge in the form of scalable QNR corpora (Section 9) could provide foundation models in which information content is compositional (Section 4.3) and substantially interpretable (Section 9.3.3). Questions of epistemic quality could be addressed by QNR-domain reasoning about external information sources (Section 9.5).

1. For a recent example, see (Fedus, Zoph, and Shazeer 2021).

2. Devlin *et al.* (2019) and Brown *et al.* (2020)

3. Radford *et al.* (2021)

4. Even relatively scalable Transformer architectures (Beltagy, Peters, and Cohan 2020; Katharopoulos *et al.* 2020; Zaheer *et al.* 2020) attend only to sections of text, not literatures.

5. In which knowledge representation and inference mechanisms entangle error-prone arithmetic and information retrieval with fluent multilingual translation.

4 Language, Cognition, and Neural Representations

Using NL as a motivation and point of departure for NL$^+$ motivates a review of its roles in communication, cognition, and the growth of human knowledge. Prospects for improving compositionality through QNR representations are key considerations.

Humans accumulate and share information through natural language, and language is woven into the fabric of human cognition. The expressive scope of NL, though limited, is unique and vast. If we seek to build artificial systems that match or exceed human cognitive capacity, then pursuing machine-oriented NL-like functionality seems necessary. The present section reviews the power and shortcomings of natural and formal languages, and from this perspective, considers prospects for quasilinguistic constructs more powerful and closer to cognitive representations than NL itself.

The expressive capacity of NL has resisted formalization, and despite its familiarity, remains poorly understood. Accordingly, in seeking more powerful representations, the proposed strategy will be to upgrade the expressive capacity of the relatively well understood linguistic *components* of language—lexical units and means for composing them—and to thereby upgrade the less well understood whole.

4.1 Language, Cognition, and Non-Linguistic Modalities

Natural language, human cognition, and the social accumulation of knowledge are deeply intertwined. Prospects for NL$^+$ systems parallel the role of NL in supporting both reasoning and the growth of knowledge.

4.1.1 The Roles and Generality of Language

Through biology and culture, natural language evolved to exploit animal vocalizations, an acoustic channel that transmits information between neural systems, constrained by limitations (working memory, processing speed) of the cognitive mechanisms that encode and decode meaning. At a societal level, sequences of symbols—written language—encode and extend speech, while at a neurological level, the semantic structures of language mesh with cognition.

Natural language has evolved under pressures toward comprehensive expressive capacity, yet its shortcomings are real and pervasive. We can regard

NL as both a benchmark to surpass and as a template for representational architectures.

4.1.2 Complementary, Non-Linguistic Modalities

Not words, really, better than words. Thought symbols in his brain, communicated thought symbols that had shades of meaning words could never have.

—Clifford Simak
City,[1] *1952*

The human use of complementary, non-linguistic modalities highlights limitations of NL. It is said that a picture is worth a thousand words, but it would be more true to say that images and language each can express semantic content that the other cannot. Another modality, demonstration of skills (now aided by video), is often complemented by both speech and images. Today, artifacts such as interactive computational models provide further modalities.

Like language, other modalities mesh with human cognition down to unconscious depths. Human thought relies not only on language, but on mental images, imagined physical actions, and wordless causal models. NL serves as a kind of glue between non-linguistic modalities—naming, linking, and explaining things; NL$^+$ frameworks must and can do likewise. Neural ML shows us that vector embeddings can describe much that words cannot, images and more. Expressive embeddings beyond the scope of human language can serve as integral, in some sense "word-like" parts of quasilinguistic semantic structures, stretching the concept of NL$^+$. The ability to directly reference and wrap a full range of computational objects stretches the concept further.

4.1.3 How Closely Linked are Language and Cognition?

Embeddings resemble biological neural representations more closely than words do: Embeddings and neural state vectors contain far more information than mere token identities, and both are directly compatible with neural(-like) processing. Thus, embedding-based QNRs are closer to cognitive representations than are natural languages, and presumably more compatible with (quasi)cognitive processing.[2]

1. Reprinted, Simak (2016).

2. This does *not* argue that internal cognitive representations themselves have a clean, graph-structured syntax, nor that sparse, syntax-like graphs are optimal representations for externalized QNRs. The argument addresses only properties of QNRs relative to word strings.

How deep are the connections between language and human cognition? Without placing great weight on introspective access to cognitive processes, and without attempting to resolve theoretical controversies, some aspects of the relationship between language and cognition seem clear:

- Language and cognition have co-evolved and have had ample opportunity to shape and exploit one another.
- Language is compositional, and the success of neural models that parse scenes into distinct objects and relationships (Figure 8.1) suggests that compositional models of the world are more fundamental than language itself.[1]
- The experience of trying to "put thoughts into words" (and sometimes failing) is good evidence that there *are* thoughts that are close to—yet not identical with—language; conversely, fluent conversation shows that substantial cognitive content readily translates to and from language.
- Externalization of thoughts in language can help to structure personal knowledge, while writing and reading can expand our mental capacities beyond the limits of memory.

These observations suggest that the use of linguistically structured yet quasicognitive representations could aid the development of machine intelligence by providing a mechanism that is known to be important to human intelligence. Conversely, prospects for high-level machine intelligence *without* something like language are speculative at best.

4.2 Cumulative and Structured Knowledge

With some loss of nuance, speech can be transcribed as text, and with some loss of high-level structure,[2] text can be mapped back to speech. A central concern of the present document, however, is the growth (and refinement) of accessible knowledge; today, this process relies on the development of text corpora that express (for example) science, engineering, law, and philosophy, together with literatures and histories that describe the human condition. Indeed, in this document, "natural language" tacitly refers to language captured in writing.

1. Applications of compositional neural models include not only language-linked explanation and question answering (Shi, Zhang, and Li 2019) but non-linguistic tasks such as predictive modeling of physical systems (Watters *et al.* 2017). However, to the extent that neural ML systems can display competence in compositional tasks *without* language-like internal representations, this counts against the necessity of language-like representations in cognition.

2. *E.g.*, due to working-memory constraints (Caplan and Waters 1999).

In communication, NL$^+$ aims to be more expressive than NL; in connection with cognition, NL$^+$ aims to be more directly compatible with (quasi)cognitive processing; on a global scale, NL$^+$ aims to be more effective in accumulating and applying general knowledge. The growth of NL$^+$ corpora can be cumulative and content can be structured for use.

Returning to a cognitive perspective, humans not only read and write language, but also "read and write" long-term memories. NL$^+$ content shares characteristics of both language and memory: Like text, NL$^+$ content constitutes explicit, shareable information; like long-term memory, NL$^+$-based systems can store (quasi)cognitive representations that are accessed through associative mechanisms.[1]

4.3 Compositionality in Language and Cognition

Why does the structure of language suit it to so many tasks? Links between language and cognition—their co-evolution, their close coupling in use—are part of the story, but correspondence between compositional structures in language, cognition, and the world is another, perhaps more fundamental consideration.

The case for the broad utility of NL$^+$ frameworks in AI is based in part on this correspondence:[2] Compositionality *in the world* speaks in favor of pursuing compositional representations of knowledge, situations, and actions. Likewise, *compositionality in cognition* (and in particular, deliberate reasoning) speaks in favor of strong roles for compositional representations in supporting quasicognitive processing.

4.3.1 Degrees of Compositionality

Here, a system—linguistic, cognitive, or actual—will be termed "compositional" if it can be usefully (though perhaps imperfectly) understood or modeled as consisting of sets of parts and their relationships.[3] *Useful* compositionality requires that this understanding be in some sense local, emerging from parts that are not too numerous or too remote[4] from the parts at the

1. Section 9.1.2 considers embedding-indexed QNR stores as associative memories accessed through near-neighbor lookup in semantic spaces.

2. Goyal and Bengio propose language-inspired compositionality as a key to developing systems that more closely model human-like intelligence (Goyal and Bengio 2021); see also (Y. Jiang *et al.* 2019), which makes strong claims along similar lines.

3. This is a softer criterion than that of *formal* compositionality.

4. In a sense that may be neither spatial nor temporal.

focus of attention or analysis. Hard, local compositionality in symbolic systems requires that the meaning of expressions be strictly determined by their components and syntactic structures;[1] in natural language, by contrast, compositionality is typically soft, and locality is a matter of degree. Strengthening the locality of compositionality can make linguistic representations more tractable, and is a potential direction for upgrading from NL.

4.3.2 Compositionality in Language and the World

Compositionality in language mirrors compositionality in the world—though the compositionality of the world as we see it may be conditioned by language and the structure of feasible cognitive processes. Language (and quasilinguistic systems such as mathematical notation and computer code[2]) can represent compositionality beyond narrow notions of discrete "things" and "events". Phenomena that are distributed in space and time (*e.g.*, electromagnetic waves, atmospheric circulation, the coupled evolution of species and ecosystems) can be decomposed and described in terms of distributed entities (fields, fluids, and populations) and relationships among their components. Entities themselves commonly have attributes such as mass, color, velocity, energy density, that are compositional in other ways.

Neural representation learning confirms that compositionality is more than a human mental construct. A range of successful ML models incorporate compositional priors or learn emergent compositional representations.[3] These successes show that compositional approaches to understanding the world are useful in non-human, quasicognitive systems.

Representations typically include parts with meanings that are conditioned on context.[4] In language, the meaning of words may depend not only on syntactically local context, but on general considerations such as the level of technicality of discourse, the epistemic confidence of a writer, or the field

1. Scoped binding of symbols stretches but does not break notions of syntactic locality—if locality is construed in terms of arcs in identifiable graphs, it is not constrained by syntactic distances over sequences or trees.

2. Interestingly, although computer languages are models of compositionality, fMRI studies show that reasoning about code is only weakly focused on brain regions specialized for natural language (Ivanova *et al.* 2020). This distribution of neural function supports a broad role for compositional representations in non-linguistic cognition.

3. Raposo *et al.* (2017), Watters *et al.* (2017), Battaglia *et al.* (2018), Eslami *et al.* (2018), G. R. Yang *et al.* (2019), and Bear *et al.* (2020)

4. In formal systems, definitions and bindings within a scope are examples of a precise form of contextually conditioned compositionality.

under discussion ("glass" means one thing in a kitchen, another in optics, and something more general in materials science). In images, the appearance of an object may depend on lighting, style, resolution, and color rendering, and scenes generated from textual descriptions can differ greatly based on contextual attributes like "day" or "city". Contexts themselves (as shown by these very descriptions) can to some extent be represented by NL, yet images generated by neural vision systems can be conditioned on contextual features that are substantially compositional (as shown by structure in latent spaces), and even recognizable by humans, yet not readily described by language.

All of these considerations speak to the value of compositional representations in language and beyond. Taken as a whole, these considerations suggest that quasilinguistic representations can describe features of the world that elude language based on strings of words.

4.3.3 Compositionality in Nonlinguistic Cognition

Compositionality in cognition parallels compositionality in language and in the world. When we perceive objects with properties like "relative position", or consider actions like "dropping a brick", these perceptions and cognitive models are compositional in the present sense; they are also fundamentally nonlinguistic yet often expressible in language. Thus, visual thinking (Giaquinto 2015) can be both non-linguistic and compositional; when this thinking is abstracted and expressed in diagrammatic form, the resulting representations typically show distinct parts and relationships.

4.3.4 Compositionality in Natural Language

The concept of language as a compositional system is woven through the preceding sections, but these have spoken of language as if compositionality in language were a clear and uncontroversial concept. It is not. Formal symbolic systems provide a benchmark for full compositionality, and by this standard, natural languages fall far short.[1] Formal concepts of compositionality have difficulty including contextual features like "topic" or "historical era" or "in

1. The lack of full compositionally in language has been a bitter pill for formalists, and not all have swallowed it. The Principle of Compositionality, that the meaning of a complex expression is determined by its structure and the meanings of its constituents, has been taken to apply to language; although others recognize a pervasive role for context (enriching word embeddings with contextual information has been a successful strategy in neural NLP; see Liu, Kusner, and Blunsom (2020)), some seek to apply context to determine (fully compositional) *lexical-level* meanings, which seems an arbitrary and perhaps unworkable choice. See Szabó (2017).

Oxford", and struggle with contextual modulation of meaning at the level of (for example) sentences rather than words.[1]

Neural NLP can (and to be effective, must) incorporate information that is beyond the scope of locally compositional representations of word strings. Transformer-based language models show something like understanding of text in a broad world context, yet the embodiment of that knowledge in their weights is *not obviously* compositional in its abstract structure, and *obviously not* compositional in its concrete representation. To say that the meaning of text emerges from a composition of its components—with particular Transformer computational states as one of those components—would stretch the meaning of compositionality to the breaking point.[2] To be meaningful, compositionality must be in some sense local.

Compositionality in language can be strengthened by strengthening the localization of contextual information. Quasilinguistic neural representations can contribute to this in two ways: First, by substituting descriptive vector embeddings for ambiguous words and phrases,[3] and second, by incorporating embeddings that locally summarize the meaning of a syntactically broad context (for example, a context on the scale of a book), together with embeddings that locally summarize remote context, for example, the kind referred to in expressions like "when read in the context of..."[4] The second role calls for embeddings that are not conventionally "lexical".[5]

In brief, the world, cognition, and language are substantially "compositional", and relative to natural language, quasilinguistic neural representations can improve local compositionality. As we will see, improving local compositionality can have a range of advantages.

1. "Castle Mound gives a good view of its surroundings." Is the context of this sentence tourism in present-day Oxford, or military intelligence during the Norman conquest? The nature of the view and its significance may be clear in the context of a pamphlet or book, but cannot be found in the quoted string of words.

2. The same can be said of natural language expressions in the context of human memory and neural states.

3. Note that the problem is not ambiguity *per se*, but ambiguity that is unintentional or costly to avoid. Intentional ambiguity is expressive, and (to meet benchmark criteria) must therefore be expressible in NL$^+$ (see Section 8.3.1 and Section A3.4).

4. Because expressions may appear in multiple contexts, this should be seen as information about the context of a particular *citation* or *use* of an expression.

5. As discussed in Section 8.3.5.

5 Expressive Constructs in NL and NL$^+$

NL$^+$ must subsume the functionality of NL constructs identified by linguists, and the shortcomings of those constructs suggest substantial scope for surpassing NL's expressive capacity.

The rich expressive constructs found in natural languages provide a benchmark and point of departure for the pursuit of more powerful capabilities. Considering linguistics in the context of neural ML suggests both challenges that must be addressed and challenges that can be set aside in pursuing the promise of NL$^+$. Figure 6.1 illustrates relationships among some overlapping classes of representation systems, some of which stretch the definition of "linguistic".

Appendix A3 explores further aspects of the relationship between natural language and potential QNR/NL$^+$ representations, including prospects for condensing, regularizing, and extending the scope of quasilinguistic representations.

5.1 Vocabulary and Structure in Natural Languages

What linguistic constructs enable NL to serve human purposes?[1] Those purposes are broad: Natural languages are richer than "knowledge representation languages"[2] or other formal systems to date; language can describe complex things, relationships, and situations, along with goals, actions, abstract argumentation, epistemic uncertainty, moral considerations, and more.

The proposed strategy for developing frameworks that subsume and extend NL is to upgrade representational functionality by upgrading both NL components and compositional mechanisms. Crucially, *this strategy requires no strong, formal theory of semantics, grammar, syntax, or pragmatics, and hence no coding of formal rules.* An approach that (merely) upgrades components and structure sidesteps questions that have generated decades of academic controversy and addresses instead a far more tractable set of problems.

1. Note this question does not ask how those constructs actually operate, which is a subject of ongoing controversy among linguists.

2. See Bobrow and Winograd (1977) and McShane and Nirenburg (2012). Knowledge representation languages typically attempt to build on unambiguous ontologies (Guarino 2009), yet the ability to express ambiguity is an important feature of natural languages.

5.2 Grammar and Syntax

It is widely agreed that sentences can usefully be parsed into trees[1] defined by grammars, yet there are several competing approaches.[2] In an NL$^+$ framework, explicit graphs can accommodate and generalize any choice, hence no choice need be made at the outset; further, because neural models can integrate information across extended syntactic regions, grammar-like choices of local graph structure need not strongly constrain semantic processing. Architectures with a QNR-oriented inductive bias together with neural representation learning on appropriate tasks should yield effective systems with NL$^+$ functionality (Section 10).

Section 7 explores potential vector/graph representations in greater depth, while Section 8 considers applications of vector embeddings that subsume elements of NL syntactic structure—again, not to propose or predict, but instead to *explore the potential* of learned NL$^+$ representations.

5.3 Words, Modifiers, and Lexical-Level Expressions

The role of lexical-level units in NL is subsumed by embeddings in NL$^+$ frameworks, and prospects for improving NL$^+$ expressiveness depend in part on the potential advantages of representations in continuous, structured spaces. It is easy to argue that embeddings can be as expressive as words (*e.g.*, embeddings can simply *designate* words), but a deeper understanding of their potential calls for considering words and word-like entities in the context of continuous semantic spaces.

5.3.1 Words and word-level units

When considering NL$^+$ frameworks from an NL perspective, a key question will be the extent to which multi-word expressions can be folded into compact, tractable, single-vector representations while gaining rather than losing expressive power. Two heuristics seem reliable:

1. Meanings that some languages express in a single word can be represented by a single vector.[3]

1. Or DAGs, when coreference is represented explicitly.

2. See Borsley and Börjars (2011).

3. Meanings, not words: Vector representations of words perform poorly when those words have multiple meanings, but representing meanings rather than words sidesteps this problem.

2. Sets of word-level units with related meanings correspond to sets of points clustered in a semantic space.

In the present context (and adopting an NL perspective), "lexical" (or "word-level") units will be restricted to a single noun or verb together with zero or more modifiers (*e.g.*, adjectives or adverbs),[1] and a "simple" noun (or verb) phrase will be one that cannot be decomposed into multiple noun (or verb) phrases.[2] The phrase "a large, gray, sleepy cat" is a simple noun phrase in this sense; "a cat and a dog" is not simple, but conjunctive.

As with many features of language, the single-vs.-conjunctive distinction is both meaningful and sometimes unclear: Is "spaghetti and meatballs" a single dish, or a pair of ingredients? Is "bait and switch" a deceptive strategy or a pair of actions? Is the semantic content of "ran, then halted" necessarily compound, or might a language have a verb with an inflected form denoting a past-tense run-then-halt action? Note that nothing in the present discussion hinges on the sharpness of such distinctions. Indeed, the typical flexibility and softness of mappings between meanings and representations speaks against discrete tokens and formal representations and in favor of QNR/NL$^+$ systems that can represent a softer, less formal semantics.[3]

In natural language, the meaning of "word" is itself blurry, denoting a concept that resists sharp definition. In linguistics, "morphemes" are the smallest meaningful linguistic units, and include not only (some) words, but prefixes, suffixes, stems, and the components of compound words. In morphologically rich languages, words may contain morphemes that denote case or tense distinctions that in other languages would be denoted by words or phrases. Blurriness again speaks in favor of soft representations and against linguistic models that treat "word" as if it denoted a natural kind.

5.3.2 Content word roles and representations

Linguists distinguish "content words" (also termed "open class" words) from "function" (or "closed class") words. The set of content words in a vocabulary

1. This use of "lexical" differs from a standard usage in linguistics, where to be "lexical", a phrase must have a meaning other than what its components might indicate, making the phrase itself a distinct element of a vocabulary. The phrases "on the other hand" and "cat-and-mouse game" are lexical in this sense. NLP research recognizes a similar concept, "multi-word expressions" (Constant *et al.* 2017).

2. Linguists define "simple phrases" differently.

3. Note that points in a vector space are inherently sharp, yet may be taken to represent (potentially soft) regions in a lower dimensional space (see Section 7.1.5).

is large and readily extended;[1] the set of function words (discussed below) is small and slow to change.

Content words typically refer to objects, properties, actions, and relationships. They include nouns, verbs, adjectives, and most adverbs, and they typically have more-or-less regular marked or inflected forms. The growth of human knowledge has been accompanied by the growth of content-word vocabularies.

From the perspective of NL syntax and semantics, adjectives and adverbs can be viewed as modifying associated nouns and verbs; this relationship motivates the description of the resulting phrases as word-level (or lexical) in the sense discussed above. In exploring the potential expressive capacity of NL$^+$ frameworks, will be natural to consider semantic embedding spaces that accommodate (meanings like those of) nouns and verbs together with the lexical-level refinements provided by adjectives, adverbs, markers, and inflections.[2]

One can think of content words as representing both distinctions of *kind* and differences in *properties*.[3] Numbers, animals, planets, and molecules are of distinct kinds, while magnitude, color, accessibility, and melting point correspond to differences in properties, potentially modeled as continuous variables associated with things of relevant kinds. In embedding spaces, one can think of distinctions of kind as represented chiefly by distances and clustering among vectors, and differences in properties as represented chiefly by displacements along directions that correspond to those properties.[4]

Note that this perspective (kinds $\rightarrow$ clusters; properties $\rightarrow$ displacements) is primarily conceptual, and need not (and likely should not) correspond to distinct *architectural* features of QNR/NL$^+$ systems.

1. The vocabulary of an English speaker may include 10,000 to 100,000 or more content words; different speakers may employ different blocks of specialized (*e.g.*, professional) vocabulary.

2. Note also the potential value of *explicitly compositional representations of embeddings, e.g.*, embeddings built by concatenation (Appendix A4).

3. Distinctions of kind and differences in properties differ, yet are not entirely distinct.

4. The somewhat counter-intuitive geometric properties of high dimensional spaces are relevant here (see Section 7.1.4). Note also that displacements in a particular direction need not have the same meaning in different regions of semantic space: Most properties relevant to planets differ from those relevant to music or software.

5.3.3 Function word roles and representations

Function (closed-class) words are diverse. They include coordinating conjunctions *(and, or, but…)* conjunctive adverbs *(then, therefore, however…)*, prepositions *(in, of, without…)* modal verbs *(can, should, might…)*, determiners *(this, that, my…)*, connectives *(and, or, because, despite…)*, and more (see Table A2.1).

While the set of open-class words is huge, the set of closed-class words is small—in English, only 200 or so. The vocabulary of open-class words can readily be extended to include new meanings by example and definition. Closed-class words, by contrast, typically play general or abstract roles, and linguists find that this small set of words is nearly fixed (hence "closed").[1] The still-awkward repurposing of "they" as a gender-neutral third-person singular pronoun illustrates the difficulty of expanding the closed-class vocabulary, even to fill a problematic gap—alternatives such as "ze" have failed to take hold (C. Lee 2019).

Function words that in themselves have minimal semantic content can shape the semantics of complex, content-word constructs (such as clauses, sentences, and paragraphs), either as modifiers or by establishing frameworks of grammatical or explanatory relationships. Representations that subsume NL must span semantic spaces that include function words.

The closed-class nature of NL function words suggests opportunities for enriching the corresponding semantic domains in NL$^+$. In English, for example, the ambiguity between the inclusive and exclusive meanings of "or" in English suggests that even the most obvious, fundamental—and in human affairs, literally costly—gaps in function-word vocabularies can go unfilled for centuries.[2]

1. The poverty of closed-class vocabulary is mitigated by the availability of compound function words *(at least, because of…)* that can be treated as lexical entities. See Kato, Shindo, and Matsumoto (2016).

2. The constructs "X and/or Y" and "either X or Y" can express the inclusive/exclusive distinction, yet trade-offs between precision and word economy (and the cognitive overhead of instead relying on context for disambiguation) ensure frequent ambiguity and confusion in practice. As a less trivial example, the ability to compactly express "possibly-inclusive-but-probably-exclusive-or" would be useful, and in a continuous vector space of function words, would also be natural.

5.3.4 TAM-C modifiers

Natural languages employ a range of tense, aspect, modality, and case (TAM-C) modifiers;[1] some are widely shared across languages, others are rare. In some languages, particular TAM-C modifiers may be represented by grammatical markers (a class of function words); in others, by inflections (a class of morphological features). Meanings that in English are conveyed by function words may be conveyed by inflections in other languages.[2] Sets of TAM-C overlap with closed-class adjectives and adverbs, and are similarly resistant to change.

Some TAM-C modifiers alter the meaning of lexical-level elements (both words and phrases); others operate at higher semantic levels. They can convey distinctions involving time, space, causality, purpose, evidence, grammatical roles, and more:

- *Tense and aspect* distinctions typically indicate times and time-spans relative to the present *(ran, run, running, had run, will have been running...);* see Table A2.2.
- *Modality* can indicate distinctions between questions, commands, confident statements, and possibilities, among many others; see Table A2.3.
- *Case* can indicate distinctions between (for example) grammatical roles (subject, object...), functional roles (instrumental case), states of possession (genitive case), or relative physical positions (various locative cases); see Table A2.4.

Note that many of these distinctions are particularly important to *situated* and *cooperating* agents.

TAM-C modifiers are sufficiently general and important that NLs encode them in compact lexical representations. The role of TAM-C modifiers in NL calls for similar mechanisms in NL$^+$, and—like adjectives and adverbs—TAM-C modifiers are natural candidates for folding into lexical-level vector representations (Section A3.3).[3]

1. Because tense, aspect, and modality overlap and intertwine, linguists often group them under the label "TAM"; because they also overlap with case distinctions, all their indicators (inflections, markers) will be lumped together here and simply referred to as "TAM-C modifiers" (for examples and discussion, see Appendix A2 and Section A3.3).

2. Further muddling standard linguistic distinctions, punctuation can indicate case or modality (question marks, exclamation marks) and can play roles like function words that clarify syntax (commas, semicolons, periods) or express relationships of explanation, example, or reference (colons, parentheses, quotation marks). In spoken language, verbal emphasis and paralinguistic signals play similar roles.

3. Linguists recognize a semantic space of modalities (Allan 2013).

5.4 Phrases, Sentences, Documents, and Literatures

NL$^+$ representations like those anticipated here condense (some) phrase-level meanings into single, hence syntax-free, embeddings. Within an NL$^+$ domain, these "phrase-level" meanings are *definitional*, not merely approximations of the meaning of a hypothetical phrase in NL. As outlined above, meanings of kinds that correspond to simple, lexical-level noun and verb phrases (in NL) are strong candidates for single-embedding representation, while the equivalents of noun and verb conjunctions (in NL) typically are not; nonetheless, equivalents of NL phrases of other kinds (some noun clauses?) could potentially be captured in single embeddings. The boundaries of the useful semantic scope of definitional vector embeddings are presently unclear, yet at some level between a word and a document, definitional embeddings must give way to vector-labeled graph representations.[1]

5.5 At the Boundaries of Language: Poetry, Puns, and Song

Discussions of potential NL$^+$ "expressiveness" come with a caveat: To say that a representation "is more expressive" invites the question "expressive to whom?" The meanings of NL text for human readers depend on potentially human-specific cognition and emotion, but NL$^+$ expressions cannot, in general, be read by humans—indeed, systems that "read" NL$^+$ (*e.g.*, systems for which "associative memory" means scalable search and "NL$^+$ expressions" are situated within a spectrum of QNRs) are apt to be quite unlike humans.

What might be called "outward-facing expressions"—descriptions, commands, questions, and so on—represent a kind of semantic content that may be *accessible* to human minds, but is not *specific* to them. The arguments above suggest that NL$^+$ representations can outperform NL in this role. However, NL text—and utterances—can convey not only outward-facing semantic content, but human-specific *affective* meaning, as well as NL-saturated associations, allusions, and word play. Puns and poetry translate poorly even between natural languages, and poetry merges into song which merges into pure music, far from what is usually considered semantic expression. For present purposes, these functions of text and utterances will be considered beyond the scope of "language" in the sense relevant to NL$^+$ functionality; what *can* be represented, however, is literal content (NL text, recorded sound) embedded

1. Potentially accompanied by abstractive, non-definitional embeddings (Section 8.3.4). The contours of such boundaries need not be determined by an implementation: There is no need to engineer representations that neural systems can instead learn.

in NL$^+$ *descriptions of its effects* on human minds (which are, after all, parts of the world to be described).[1]

In the context of this document, the content (or "meaning") of natural language expressions will be equated with semantic content in the outward-facing sense. When a poem delivers a punch in the gut, this is not its *meaning*, but its *effect*.

6 Desiderata and Directions for NL$^+$

> Prospects for improving expressiveness in NL$^+$ representations include mechanisms both like and beyond those found in natural languages. Research directions aimed at realizing these prospects are well-aligned with current directions in neural ML.

Building on the NL-centered considerations discussed above, and looking forward to mechanisms beyond the scope of natural language, this section outlines desiderata for NL$^+$ (and more generally, QNR) functionality. Appendix A3 further explores NL$^+$-oriented prospects for QNR frameworks.

6.1 Improve and Extend Fundamental NL Constructs

Desiderata for NL$^+$ representations include improving and extending fundamental NL constructs:

Subsume (but do not model) NL: The aim of research is not to *model NL*, but to model (and extend) *its functionality*.

Exploit deep learning: Use neural ML capabilities to extend NL constructs without hand-crafting representations.

Improve compositionality: Embeddings can provide effectively infinite vocabularies and loosen the dependence of meaning on context.

Use explicit graph representations: Graphs can compose embeddings without the constraints of NL syntax and ambiguities of coreference.

***Exploit vector* embedding spaces:** Relative to words, embeddings can improve both expressive capacity and computational tractability:

- Embeddings are natively neural representations that need not be decoded and disambiguated.

1. Philosophers have dissected considerations of this sort to provide richer distinctions and more precise terminology.

- Embeddings, unlike words, are differentiable, facilitating end-to-end training.
- Embeddings provide effectively infinite vocabularies, enriching expressive capacity.

Embrace (but do not impose) formal systems: NL$^+$ frameworks and formal systems are complementary, not competing, modes of representation. Formal systems can be embedded as sub-languages in NL$^+$, much as mathematical expressions are embedded in textbooks (Section 9.2.4).

6.2 Exploit Mechanisms Beyond the Scope of Conventional NL

The discussion above has focused on features of NL$^+$ frameworks that can be regarded as upgrades of features of NL, replacing words with embeddings and implicit syntactic trees with explicit, general graphs. There are also opportunities to exploit representations beyond the scope of NL: These include vector representations that enable novel, high-level forms of expression, abstraction, and semantic search, as well as QNR-based tools for knowledge integration at scale syntactically embedded non-linguistic content far beyond the bounds of what can be construed as language (Section 9.2).

6.2.1 Use Embeddings to Modify and Abstract Expressions

Embeddings can perform semantic roles at the level of sentences, paragraphs, and beyond. In an adjective-like role Section 8.3.1), they can modify or refine the meaning of complex expressions; in an abstractive role, they can enable efficient, shallow processing (skimming) (Section 8.3.4).

6.2.2 Use Embeddings to Support Scalable Semantic Search

Abstractive embeddings can support similarity-based semantic search—in effect, associative memory—over NL$^+$ corpora. Efficient similarity search scales to repositories indexed by billions of embeddings (Section 9.1.5).

6.2.3 Reference, Embed, and Wrap Everything

At a syntactic level, NL$^+$ frameworks can embed not only formal systems, but also content of other kinds (Figure 6.1):

Non-linguistic lexical-level units: Neural embeddings can represent objects that differ from words, yet can play a similar role. For example, image embeddings can act as "nouns", while vector displacements in a latent space can act as "adjectives" (Section 8.2).

Non-linguistic objects: Through hyperlinks, online NL expressions can in effect incorporate arbitrary informational or computational objects. NL$^+$ expressions can do likewise (Section 9.2).

Linguistic objects: NL$^+$ expressions can reference, describe, and help index linguistic and language-infused objects such as books, websites, and video. NL$^+$ content can also cite sources (Section 9.5.2).

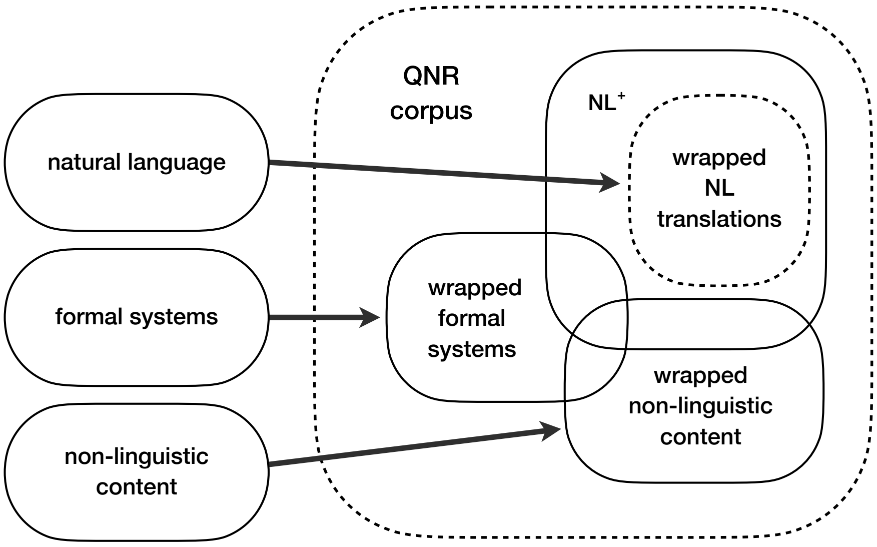

Figure 6.1: *Approximate transformation and containment relationships among representation systems.*

6.3 Exploit QNRs to Support Knowledge Integration

NL$^+$ representations can support tools for knowledge integration based on semantic search over QNR corpora in conjunction with soft matching, unification, and generalization over QNR representations.[1] These operations can compare and combine expressions to identify and represent areas of concordance or conflict, as well as structural analogies, pattern completions, and

1. Unification of two expressions produces the least specific (most general) expression that contains the information of both; unification fails if expressions contain conflicting information. Generalization (anti-unification) of two expressions produces the most specific (least general) expression that is compatible with (and hence unifies) both. Unification and anti-unification are associative and commutative, and satisfy several other axioms. Soft unification and anti-unification relax these constraints. (See Section A1.4.3.)

semantic overlaps that enable the integration of narrow expressions to build semantic structures span broader domains. Soft unification of QNR structures can support continuous relaxations of logical inference through (for example) Prolog-like computation (Section A5.1). Thus, QNR representations and corpora can support knowledge integration through mechanisms beyond those readily available in processing NL text.

6.4 Build on Current Research

NL$^+$-oriented research can build on current ML applications, methods, architectures, training data, and tool sets. Potentially useful components include Transformers, graph neural networks, models that embed or generate graph/vector representations, and multitask learning methods that can be applied to shape multipurpose representations.

NL$^+$ representations and corpora have natural applications in translation, question answering, and conversational interfaces, as well as reinforcement learning (RL).[1]

NL$^+$-enabled systems could contribute to current research objectives in mathematics, science, engineering, robotics, and machine learning itself, both by helping to integrate and mobilize existing knowledge (*e.g.*, mining literatures to identify capabilities and opportunities), and by facilitating research that produces new knowledge. Efforts to harness and extend the power of natural language align with aspirations for advanced machine learning and artificial intelligence in general.

6.5 Some Caveats

The present discussion describes general frameworks, mechanisms, and goals, but the proposed research directions are subject to a range of potential (and equally general) criticisms:

- Proposed NL$^+$ frameworks are templated on NL, but perhaps too closely to provide fundamentally new capabilities.
- Alternatively, proposed NL$^+$ frameworks may differ too greatly from NL, undercutting the feasibility of equaling NL capabilities.
- In light of the surprising power of flat neural representations, inductive biases that favor QNRs might impede rather than improve performance.

1. For applications of language in RL, see Das *et al.* (2017), Lazaridou, Peysakhovich, and Baroni (2017), Shah *et al.* (2018), and Luketina *et al.* (2019).

- Both neural ML and human cognition embrace domains that are decidedly non-linguistic, limiting the scope of NL-related mechanisms.
- Ambitious NL$^+$ applications may call for more semantic structure than can readily be learned.
- Implementation challenges may place ambitious NL$^+$ applications beyond practical reach.
- Current research may naturally solve the key problems with no need to consider long-term goals.
- The prospects as described are too general to be useful in guiding research.
- The prospects as described are too specific to be descriptive of likely developments.

Most of these criticisms are best regarded as cautions: Linguistic mechanisms have limited scope; relaxing connections to NL may improve or impede various forms of functionality; implementations of working systems can be difficult to develop or fall short of their initial promise; motivations and outlines of research directions are inherently general and open-ended; generic, short-term motivations often suffice to guide developments up a gradient that leads to capabilities with far-reaching applications.

Nonetheless, despite these caveats, near-term research choices informed by QNR/NL$^+$ concepts seem likely to be more fruitful than not, leading to tools and insights that enable and inform further research. Much current research is already well-aligned with QNR development and NL$^+$ aspirations, and it is interesting to consider where that research may lead.

7 Vector-Labeled Graph Representations

In conjunction with today's deep learning toolkit, vector-labeled graph representations provide powerful, differentiable mechanisms for implementing systems that represent and process structured semantic information.

This present section examines vector-labeled graph representations (VLGs) from the perspectives of representational capacity and neural ML tools; the following section will examine prospects for applying this representational capacity to implement QNR systems that surpass the expressive capacity of natural language.[1]

7.1 Exploiting the Power of Vector Representations

In a sense, the large representational capacity of typical high-dimensional embedding vectors is trivial: Vectors containing hundred or thousands of floating point numbers contain enough bits to encode lengthy texts as character strings. What matters here, however, is the representational capacity of vectors in the context of neural ML—the scope and quality of representations that can be discovered and used by neural models that are shaped by suitable architectural biases, loss functions, and training tasks. This qualitative kind of capacity is difficult to quantify, but examples from current practice are informative.

7.1.1 Vector Representations are Pervasive in Deep Learning

Deep learning today is overwhelmingly oriented toward processing continuous vector representations, hence the extraordinary capabilities of deep learning testify to their expressive power.

7.1.2 Vector Representations Can Encode Linguistic Semantic Content

Continuous vector representations in NLP shed light on prospects for expressive, tractable QNRs.[2]. The two leading roles for vector embeddings in proposed QNR systems are (1) definitional representations of lexical-level components (paralleling vector semantics in NL and word embeddings in NLP, Section 8.2) and (2) abstractive representations of higher-level constructs for indexing and summarization (Section 8.3.4).

7.1.3 Single Vectors Can Serve as Compositional Representations

In conventional symbolic systems, compositionality enables complex meanings to be represented by combinations of components. In vector representations, meanings can be attributed to orthogonal vector components (*e.g.*, representing different properties of something), then those components can

1. These representations can also be approximately as compact as NL text (see Appendix A5).

2. Note, however, successful applications of *discretized* representations in which learned, finite sets of vectors are selected from continuous spaces; see, for example, Oord, Vinyals, and Kavukcuoglu (2018) and Razavi, Oord, and Vinyals (2019)

be combined by vector addition and recovered by projection onto their corresponding axes. Condensing what in NL would be multi-component, lexical-level syntactic structures into single embeddings can reduce the number of distinct representational elements, retain semantic compositionality, and enable facile manipulation by neural computation.

7.1.4 High-Dimensional Spaces Contain Many Well-Separated Vectors

In considering the representational capacity of high-dimensional vectors, it is important to recognize ways in which their geometric properties differ from those of vectors in the low-dimensional spaces of common human experience. In particular, some measures of "size" are exponential in dimensionality, and are relevant to representational capacity.

Call a pair of unit-length vectors with cosine similarity ≤ 0.5 "well separated". Each of these vectors defines and marks the center of a set (or "cluster") of vectors with cosine similarity ≥ 0.86; these sets are linearly separable and do not overlap. How many such well-separated cluster-centers can be found in a high-dimensional space?

In a given dimension d, the number of vectors $k(d)$ that are well separated by this criterion is the "kissing number", the maximal number of non-overlapping spheres that can be placed in contact with a central sphere of equal radius (Figure 7.1). Kissing numbers in low-dimensional spaces are small ($k(2) = 6$, $k(3) = 12\ldots$), but grow rapidly with d. For $d = 64$ and 128,[1] $k(d) > 10^7$ and 10^{12}; for $d = 256$, 512, and 1024, an asymptotic lower bound (Edel, Rains, and Sloane 2002) $k(d) > 2^{0.2075\ldots d}$ gives $k(d) > 10^{14}$, 10^{30}, and 10^{62}. Thus, the number of neighboring yet well-separated cluster-centers that can be embedded in spaces of dimensionalities commonly used in neural ML is far (!) in excess of any possible NL vocabulary.[2]

Note that the region around a cluster-center itself has great representation power for sub-clusters: For example, its content can be separated from the rest of the space by a linear threshold operation and then scaled and projected into a space of d–1 dimensions, where similar considerations apply recursively

1. Cited in Torquato and Stillinger (2006).

2. Note that the number of vectors that are all nearly orthogonal (all pairwise cosine similarities $\ll 0.5$) also grows exponentially with d and becomes enormous in high dimensional spaces.

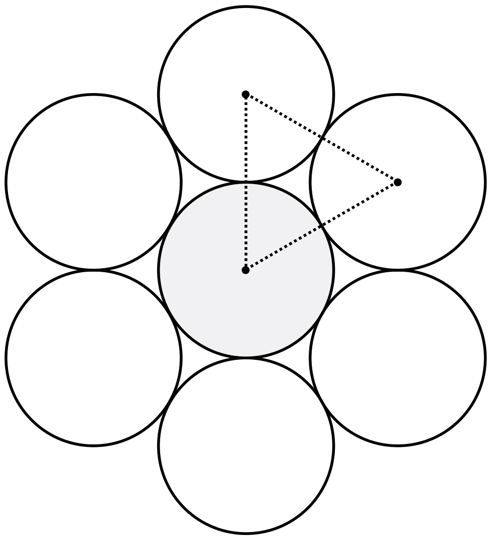

Figure 7.1: *Kissing spheres, $d = 2$, $k = 6$*

so long as the residual dimensionality remains large.[1]

The above description is intended to provide an intuition for some aspects of the expressive capacity of high-dimensional vector spaces, not to predict or suggest how that capacity will or should be used in prospective systems: Learned representations in current neural ML may offer a better guide.

7.1.5 Points Can Represent Regions in Lower-Dimensional Spaces

https://www.overleaf.com/project/61183c4a1765f22abf2e3ebf An embedding of dimensionality $2d$ can represent (for example) a center-point in a d-dimensional semantic space together with parameters that specify a surrounding box in that space.[2] An embedding may then designate, not a specific meaning, but a range of potential meanings; alternatively, a range can be regarded as a specific meaning in a semantic space that explicitly represents ambiguity. Interval arithmetic[3] generalizes some operations on d-dimensional points to operations on d-dimensional boxes.

1. The use of Euclidean distance or cosine similarity is explicitly or tacitly assumed in much of this document, but growing interest suggests that hyperbolic spaces (useful in graph and sentence embeddings) or mixed geometries may provide attractive alternatives for embedding QNR expressions; see for example Peng *et al.* (2021).

2. Appendix A1, Section A1.5.3, and Section A1.6 discuss both boxes and more flexible classes of representations in the context of unification and generalization operations on soft lattices.

3. Arithmetic in which operands are intervals over $\mathbb{R}$.

This document will usually discuss embeddings as if they represent points in semantic spaces, with operations on embeddings described as if acting on vectors that designate points, rather than regions. The concept of semantic regions becomes central, however, in considering semantic lattice structure and constraint-based inference that generalizes logic programming (Section A1.4).

7.2 Exploiting the Power of Graph-Structured Representations

Graphs are ubiquitous as representations of compositional structure because they can directly represent things and their relationships as nodes and arcs. Graphs (in particular, trees and DAGs) are central to traditional, word-oriented natural language processing, while general graphs have found growing applications in diverse areas of neural ML. This section outlines several classes of graphs and their potential roles in representing and processing semantic information.

7.2.1 Terminology, Kinds, and Roles of Graphs

The components of graphs are variously (and synonymously) called *edges, arcs, or links* (potentially directed), which connect *vertices, points, or nodes,* which in turn may carry *labels*, *attributes*, or *contents.* The present discussion will typically refer to *arcs* (or *links* between document-scale objects) that connect *nodes* that carry *attributes* or *labels* (in a semantic context, *contents* or *embeddings*).[1]

Here, "graph" typically denotes a directed graph with attribute-bearing nodes. Labeled arcs, multigraphs, and hypergraphs[2] are potentially useful but not explicitly discussed; weighted arcs are essential in some differentiable graph representations. (Sequences of embeddings can be viewed as instances of a particularly simple class of graphs.)

In prospective QNR frameworks, vector-labeled graphs have at least two areas of application: The first area parallels the use of graphs in classic NLP, where expressions are typically parsed and represented as syntax trees or (to

1. In some formal models, arcs also carry attributes. Without loss of generality, graphs G with labeled arcs can be represented by bipartite graphs G' in which labeled arcs in G correspond to labeled nodes in G'. For the sake of simplicity (and despite their potential importance) the present discussion does not explicitly consider labeled arcs. In general, computational representations of graphs will be implementation-dependent and will change depending on computational context (*e.g.*, soft, internal representations in Transformers translated to and from hard, external representations in expression-stores).

2. Van Lierde and Chow (2019) and Menezes and Roth (2021)

represent resolved coreferences) DAGs; VLGs can represent syntactic trees explicitly, bypassing parsing, and can represent coreference through DAGs, bypassing resolution. The second area of application involves higher-level semantic relationships that in NL might be represented by citations; in an NL$^+$ context, similar relationships are naturally represented as general, potentially cyclic graphs. (These topics are discussed in Section 8.1.)

7.2.2 VLGs Can Provide Capacity Beyond Stand-Alone Embeddings

Fixed-length embeddings lack scalability, while sequences of embeddings (*e.g.*, outputs of Transformers and recurrent networks), though potentially scalable, lack explicit, composable, and readily manipulated structure.

Arcs, by contrast, can compose graphs to form larger graphs explicitly and recursively with no inherent limit to scale or complexity, and subgraph content can be referenced and reused in multiple contexts. Trees and graphs are standard representations for compositional structure in a host of domains, and are latent in natural language. Graphs with vector attributes can expand the representational capacity of sets of embeddings by placing them in a scalable, compositional framework.[1]

7.2.3 VLGs Can Be Differentiable

Typical neural operations on vector attributes are trivially differentiable, while differentiable operations on representations of graph topologies require special attention. Without employing differentiable representations, options for seeking graphs that minimize loss functions include search (potentially employing heuristics or reinforcement learning) and one-shot algorithmic construction.[2] With differentiable representations, more options become available, including structure discovery through end-to-end training or inference-time optimization.

Conventional representations in which nodes and arcs are simply present or absent can be termed “hard graphs”; representations can be made “soft” and differentiable by assigning weights in the range [0, 1] to arcs. Differentiable algorithms that assign weights tending toward {0, 1} can recover conventional

1. Single embeddings can, however, represent small graphs; thus, graph-structured representation does not always require reification of nodes and arcs (See Section 7.3).

2. Yun *et al.* (2019), J. Zhou *et al.* (2020), Kazi *et al.* (2020), and Wu *et al.* (2021). For applications of RL to graph construction, see Z. Zhou *et al.* (2019) and Trivedi, Yang, and Zha (2020).

graphs by discarding edges when their weights approach zero, implementing structure discovery through differentiable pruning.

Differentiable pruning operates on a fixed set of nodes and typically considers all pairs, impairing scalability,[1] but algorithms that exploit near-neighbor retrieval operations on what may be very large sets of nodes (Section 9.1.5) could implement scalable, differentiable, semantically informed alternatives that do not *a priori* restrict potential topologies. In typical graph representations, a link is implemented by a semantically meaningless value (*e.g.*, an array index or hash-table key) that designates a target node. Through near-neighbor retrieval, by contrast, a vector associated with a source-node can serve as a query into a semantically meaningful space populated by keys that correspond to candidate target-nodes. Selecting the unique node associated with the nearest-neighbor key yields a hard graph; attending to distance-weighted sets of near neighbors yields a soft graph.[2]

In this approach, query and key embeddings can move through their joint embedding space during training, smoothly changing neighbor distances and the corresponding arc weights in response to gradients. Mutable, soft-graph behavior can be retained at inference time, or models can output conventional, hard-graph VLGs, potentially retaining geometric information. Thus, models that build and update QNR corpora could provide fluid representations in which changes in embeddings also change implied connectivity, implicitly adding and deleting (weighted) arcs. In semantically structured embedding spaces, the resulting changes in topology will be semantically informed.

Weighted graphs may also be *products* of a computation rather than *intermediates* in computing hard-graph representations. Substituting a set of weighted arcs for a single hard arc could represent either structural uncertainty (potentially resolved by further information) or intentional, semantically informative ambiguity.

7.2.4 VLGs Can Support Alignment and Inference

Neural algorithms over VLGs can support both processing within single expressions and operations that link or combine multiple expressions. These

1. To enable arbitrary graphs as outputs requires initializing with complete graphs (hence $N(N-1)$ directed arcs). Restricting optimization to local subgraphs, to restricted search spaces, or to algorithmically generated "rough drafts" can avoid this difficulty.

2. Note, however, that efficient, highly scalable near-neighbor retrieval algorithms on unstructured spaces are typically "approximate" in the sense that nearest neighbors may with some probability be omitted. This shortcoming may or may not be important in a given application, and fast algorithms on weakly structured spaces can be exact (see Lample *et al.* 2019).

have potentially important applications within QNR frameworks.

Algorithms that align similar graphs[1] can facilitate knowledge integration over large corpora, supporting the identification of clashes, concordances, and overlaps between expressions, and the construction of refinements, generalizations, analogies, pattern completions, and merged expressions,[2] as well as more general forms of interpretation and reasoning. Differentiable algorithms for graph alignment include continuous relaxations of optimal assignment algorithms that enable end-to-end learning of alignable, semantically meaningful embeddings (Sarlin *et al.* 2020).

7.2.5 VLGs Can Support (Soft) Unification and Generalization

If we think of expressions as designating regions of some sort[3], then to *unify* a pair of expressions is to find the largest expressible intersection of their regions, while to *anti-unify* (or *generalize*) a pair of expressions is to find the narrowest expressible union of their regions. Domains that support these operations (and satisfy a further set of axioms) constitute mathematical lattices.[4] Given a set of expressions, unification may be used to infer narrower, more specific expressions, while anti-unification operations may be used to propose broader, more general expressions.

As discussed above, QNR expressions that represent explicit uncertainty or ambiguity (*e.g.*, containing vectors representing uncertain or partially constrained values) may be regarded as representing regions in a semantic space. The nature of "expressible regions" in the above sense depends on choices among representations.

Notions of "soft" or "weak" unification (discussed in Section A1.4.3) can replace *equality* of symbols with *similarity* between point-like semantic embeddings, or *approximate overlap* when embeddings are interpreted as representing semantic regions. Soft unification supports continuous, differentiable relaxations of the Prolog backward-chaining algorithm, enabling soft forms of multi-step logical inference. Applications of soft unification include question

1. Heimann *et al.* (2018), Cao *et al.* (2019), Qu, Tang, and Bengio (2019), and Fey *et al.* (2020)

2. Soft unification and anti-unification operations can contribute to this functionality (Appendix A1).

3. In logic, symbols correspond to zero-volume regions, while variables correspond to unbounded regions.

4. See Section A1.2. In generalizations of lattice representations, a "region" may be fuzzy, in effect defining a pattern of soft attention over the space, and lattice relationships may be approximate (Section A1.4.3).

answering, natural-language reasoning, and theorem proving;[1] Soft operations can also infer variables from sets of values (Cingillioglu and Russo 2020). As noted above, unification and generalization have further potential applications to knowledge integration in NL$^+$ corpora.

7.3 Mapping Between Graphs and Vector Spaces

Research has yielded a range of techniques for mapping graphs to and from vector representations. These techniques are important because they bridge the gap between high-capacity compositional structures and individual embeddings that lend themselves to direct manipulation by conventional (non-GNN) neural networks.

7.3.1 Embeddings Can Represent and Decode to Graphs

Neural models can be trained to encode graphs[2] (including tree-structured expressions[3]) as embeddings, and to decode embeddings to graphs.[4] A common training strategy combines graph-encoding and generative models with an autoencoding objective function.[5]

In the domain of small graphs, embeddings could encode representations with considerable accuracy—potentially with definitional accuracy, for graphs that are small in both topology and information content. Note that summary, query, and key embeddings (Section 8.3.4 and Section 9.1.2) can represent semantic content without supporting graph reconstruction.

7.3.2 Embeddings Can Support Graph Operations

Embeddings that enable accurate graph reconstruction provide differentiable graph representations beyond those discussed in Section 7.2.3. Whether directly or through intermediate, decoded representations, graph embeddings can support a full range of VLG operations. Accordingly, this class of embeddings can be considered interchangeable with small[6] VLGs, and need not be

1. Rocktäschel and Riedel (2017), Campero *et al.* (2018), Minervini *et al.* (2018), Weber *et al.* (2019), and Minervini *et al.* (2020)

2. Narayanan *et al.* (2017), Grohe (2020), and Pan *et al.* (2020)

3. Allamanis *et al.* (2017), R. Liu *et al.* (2017), H. Zhang *et al.* (2018), and Alon *et al.* (2019)

4. Y. Li *et al.* (2018) and Cao and Kipf (2018)

5. Pan *et al.* (2018), Simonovsky and Komodakis (2018), and Salehi and Davulcu (2020)

6. What counts as “small” is an open question.

considered separately here. Potential advantages include computational efficiency and seamless integration with other embedding-based representations.

8 Quasilinguistic Neural Representations

Applications of vector-labeled graphs can generalize NL syntax and upgrade NL words to implement quasilinguistic neural representations that parallel and surpass the expressive capacity of natural language at multiple levels and scales.

The present section discusses how vector-labeled graphs (VLGs) can be applied to implement quasilinguistic neural representations (QNRs) that improve on natural languages by upgrading expressive capacity, regularizing structure, and improving compositionality to facilitate the compilation, extension, and integration of large knowledge corpora. Appendix A3 covers an overlapping range of topics in more detail and with greater attention to NL as a model for potential NL$^+$ frameworks.

As usual in this document, *conceptual* features and distinctions should not be confused with *actual* features and distinctions that in end-to-end trained systems may be neither sharp, nor explicit, nor (perhaps) even recognizable. Conceptual features and distinctions should be read neither as proposals for hand-crafted structures nor as confident predictions of learned representations. They serve to suggest, not the concrete form, but the potential scope of representational and functional capacity.

8.1 Using Graphs as Frameworks for Quasilinguistic Representation

Proposed QNRs are vector-labeled graphs.[1] Attributes can include type information as well as rich semantic content; to simplify exposition,[2] attributes of what are semantically arcs can be regarded as attributes of nodes in a bipartite graph,[3] and will not be treated separately. Like trees, general (*e.g.*, cyclic) graphs can have designated roots.

1. Attributes (labels) can potentially be expanded to include sets of vectors of explicitly or implicitly differing types.

2. And perhaps also implementation.

3. A representation that can also accommodate multigraphs and hypergraphs.

8.1.1 Roles for Graphs in NL-Like Syntax

As already discussed, vector attributes can represent lexical-level components, while graphs can represent their syntactic compositions; where syntax in NL is implicit and often ambiguous, QNR-graphs can make syntactic relationships (and in particular, coreference) explicit, thereby disentangling these relationships from lexical-level semantic content. (See also Section A3.1.1.)

8.1.2 Roles for Graphs Beyond Conventional Syntax

In an extended sense, the syntax of NL in the wild includes the syntax of (for example) hierarchically structured documents with embedded tables, cross references, and citations.[1] QNRs can naturally express these, as well as syntactic structures that are unlike any that can be displayed in a document.[2]

Stretching the concept of syntax, graphs can express networks of relationships among objects. Scene graphs provide a concrete illustration of how recognizably linguistic relationships can naturally be expressed by a non-conventional syntax (for a simple example, see Figure 8.1).

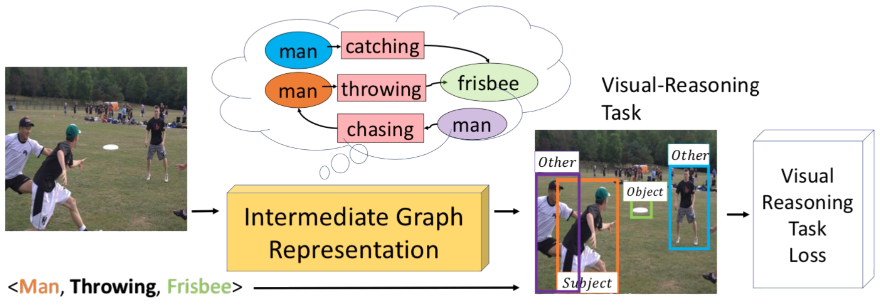

Figure 8.1: *A neurally inferred scene graph that relates several entities through subject–verb–object relationships. An enriched graph for this scene could represent more complex relationships (e.g., man1 failing to prevent man2 from throwing a Frisbee to man3). An enriched representation of entities could replace instances of labels like "man" with embeddings that have meanings more like* man-with-appearance(x)-posture(y)-motion(z), *replace instances of verbs like "catching" with embeddings that have meanings more like* probably-will-catch-intended-pass, *and so on.*[3]

1. And footnotes.
2. Examples not included.
3. Figure from Raboh *et al.* (2020), used with permission.

8.2 Using Embeddings to Represent Lexical-Level Structure

At a lexical level, embedding-space geometry can contribute to expressive power in several ways:

Proximity can encode semantic similarity.
Large displacements can encode differences of kind.
Small offsets (or larger displacements in distinguishable subspaces) can encode graded, multidimensional semantic differences.

Where NL depends on context to disambiguate words, QNRs can employ lexical-level embeddings to express meanings that are substantially disentangled from context. (For further discussion, see Section A3.2.1.)

8.2.1 Proximity and Similarity

Neural representation learning typically places semantically similar entities near one another and unrelated entities far apart. Embedding NL words works well enough to be useful, yet its utility is impaired by polysemy and other ambiguities of natural language.[1] By construction, native embedding-based representations minimize these difficulties.

8.2.2 Graded Semantic Differences

The direction and magnitude of incremental displacements in embedding spaces can represent graded, incremental differences in properties among similar entities: The *direction* of a displacement axis encodes the *kind* of difference, the *magnitude* of the displacement along that axis encodes the *magnitude* of the corresponding difference,[2] and the (commutative) addition of displacements composes multiple differences without recourse to syntactic constructs.

1. When a single word (*e.g.*, "match") has multiple unrelated meanings (homonymy), it corresponds to multiple, potentially widely scattered points in a natural semantic space; when a word (*e.g.*, "love") has a range of related meanings (polysemy) representations that map word-meanings to points (rather than semantic regions) become problematic. See Vicente (2018).

2. When different regions of a vector space represent things of distinct kinds, the meanings of directions may vary with position: In other words, because different kinds of things have different kinds of properties, it is natural for mappings of directions to properties to be a function of kinds, and hence of location. Indeed, the literature describes discrete-word models of NL semantics in which the meanings of adjectives are a function of their associated nouns; see, for example, Baroni and Zamparelli (2010) and Blacoe and Lapata (2012). In this general approach, the meanings of verbs are also a function of their associated nouns, and the meanings of adverbs are functions of their associated verbs.

By avoiding difficulties arising from the discrete words and word-sequences of NL, the use of continuous, commutative vector offsets can substantially regularize and disentangle lexical-level semantic representations.

8.2.3 Relationships Between Entities of Distinct Kinds

Distinctions between entities of different kinds[1] can be represented as discrete displacement vectors, which can encode degrees of similarity in distances and cluster things of similar kinds (animals with animals, machines with machines, *etc.*, as seen in the geometry of word embeddings).

Displacement *directions* can also encode information about kinds. Word embeddings trained only on measures of co-occurrence in text have been found to represent relationships between entities in displacement vectors, and analogies in vector differences and sums (Allen and Hospedales 2019). In neural knowledge graphs,[2] relationships between entities can be encoded in geometries that are deliberately constructed or induced by learning.[3] Lexical-level QNR labels can provide similar representational functionality.

8.3 Expressing Higher-Level Structure and Semantics

The discussion above addressed the vector semantics of embeddings that play lexical-level roles (*e.g.*, nouns and verbs with modifiers); the present discussion considers roles for embeddings in the semantics of higher-level (supra-lexical) QNR expressions. Some of these roles are *definitional:* in these roles, embeddings modify the meanings of higher-level constructs.[4] In other roles, embeddings are *abstractive:* Their semantic content may corresponds to (quasi)cognitive results of reading higher-level constructs, and are therefore semantically optional (in effect, they cache and make available computational results).

As usual, the aim here is to describe *available representational functionality*, not to predict or propose specific representational architectures. Representa-

1. Where "entity" is intended to include (at least) different things (horses, cows, photons, telescopes, gravitation, integers) and different actions (walk, run, observe, report).

2. "[A] graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities" (Hogan *et al.* 2021).

3. Ji *et al.* (2021) and Ali *et al.* (2021)

4. Expression-level definitional embeddings are not sharply distinguished from lexical embeddings, *e.g.*, those corresponding to function words.

tion learning within a QNR framework need not (and likely will not) respect these conceptual distinctions.

8.3.1 Expression-Level Modifiers (Definitional)

Although lexical-level embeddings can condense (analogues of) phrases that include modifiers of (analogues of) words, some lexical-level modifiers operate on supra-lexical expressions that cannot be condensed in this way. These modifiers may resemble adjectives, but are attached to units more complex than words. Epistemic qualifiers (Section A3.4) provide examples of this functionality.

Syntactically, definitional modifiers of this kind could be applied to expressions as vector attributes of their root nodes; semantically, expression-level modifiers act as functions of a single argument: an expression as a whole.

8.3.2 Expression-Level Relationships (Definitional)

In NL, conjunctive elements (often function words) can compose and modify the meanings of combined expressions. Some can operate on either lexical-level units or complex expressions (examples include *and*, *or*, *and/or*, *both-and*); others (for example, *however*, *because*, and *despite*[1]) typically compose meaning across substantial syntactic spans, operating not only at the level of clauses and sentences, but at the level of paragraphs and beyond.

Syntactically, a relationship between expressions could be applied through vector attributes of the root node of a composition of syntactic structures (subgraphs of a tree, DAG, *etc.*); semantically, expression-level relationships correspond to functions of two or more arguments: the expressions they relate.

8.3.3 Expression-Level Roles (Definitional)

Expressions and the things they describe have roles (frequently with graded properties) in larger contexts; features of roles may include purpose, importance, relative time, epistemic support, and so on. As with expression-level modifiers and relationships, role descriptions of this kind could be applied to an expression through vector attributes of its root node.

In general, however, a role may be specific to a *particular* context. If an expression is contained (referenced, used) in multiple contexts, it may have multiple roles, hence representations of those roles must syntactically stand

1. See also: punctuation.

above root-nodes and their attributes. More concretely, multiple contexts may link to an expression through nodes that represent the corresponding contextual roles.

8.3.4 Content Summaries (Abstractive)

In human cognition, the act of reading a text yields cognitive representations—in effect, abstractive summaries—that can contribute to understanding directly, or by helping to formulate (what are in effect) queries that directly enable retrieval of relevant long-term memories, or indirectly enable retrieval of relevant texts.[1] In a QNR context, dynamically generated summary embeddings can play a similar, direct role at inference time. When stored, however, summaries can do more:

- Amortize reading costs for inference tasks.
- Enable efficient skimming.
- Stretch the semantic span of attention windows.[2]
- Provide keys for scalable associative memory.

The nature of a summary may depend on its use, potentially calling for multiple, task-oriented summaries of a single expression.[3]

Semantically, content summaries are not expression-level modifiers: Summaries *approximate* semantics, but modifiers help *define* semantics. Content summaries may depend on contextual roles, placing them in syntactic positions above the roots of summarized expressions. A natural auxiliary training objective would be for the similarity of pairs of content-summary vectors to predict the unification scores (Section A1.4.3) of the corresponding pairs of expressions.[4]

8.3.5 Context Summaries (Abstractive)

Context summaries (*e.g.*, embeddings that summarize context-describing QNRs) represent information that is potentially important to interpreting and

1. In processing with access to QNR corpora, the analogues of long-term memory and text converge.

2. For an analogous application of summarization in Transformers, see Rae *et al.* (2019).

3. *E.g.*, representations of substantive content (potential answers to questions) *vs.* representations of the *kinds of questions* that the substantive content can answer (potentially useful as keys).

4. Summaries with this property seem well suited for use as keys in search guided by semantic similarity.

using a QNR expression.[1]

- What are the general and immediate topics of the context?
- Is the context a formal or an informal discussion?
- How does the context use the expression?
- Does the context support or criticize the content of the expression?

At inference time, context summaries could direct attention to relevant commonsense knowledge regarding a particular field or the world in general.[2] Interpreting an expression correctly may require contextual information that is distributed over large spans; providing (quasi)cognitive representations of contexts can make this information more readily available. Like content summaries, context summaries are semantically optional, but the potential scale and complexity of contexts (local, domain, and global) may make some form of context-summarization unavoidable, if extended contexts are to be used at all. Syntactically (and to some extent semantically), context summaries resemble contextual roles.

8.3.6 Origin Summaries (Abstractive)

Origin summaries (potentially summarizing origin-describing QNRs) can indicate how an expression was derived and from what information:

- What inputs were used in constructing the expression?
- What was their source?
- What was their quality?
- What inference process was applied?

A structurally distinct QNR expression can be viewed as having a single source (its "author"), and hence its root node can be linked to a single origin summary. Origins and their summaries are important to the construction, correction, and updating of large corpora.

1. The considerations here are quite general; for example, context representations are also important in understanding/recognizing objects in scenes (Tripathi *et al.* 2019; Carion *et al.* 2020).

2. Standard Transformers represent a span of immediate context in their activations, while representing both relevant and irrelevant global context in their parameters. Both of these representations are subject to problems of scaling, cost, compositionality, and interpretability. Models (potentially Transformer-based) that write and read QNRs could benefit from externalized representations of wide-ranging, semantically indexed contextual knowledge. The application of very general knowledge, however, seems continuous with interpretive skills that are best embodied in model parameters.

8.4 Regularizing, Aligning, and Combining Semantic Representations

Disentangling and regularizing semantic representations—a theme that runs through Section 8.1 and Section 8.2—has a range of benefits. In conjunction with basic QNR affordances, regularization can facilitate the alignment of representations, which in turn can facilitate systematic inference, comparison, combination, and related forms of semantic processing at the level of collections of expressions.

8.4.1 Regularizing Representations

In natural languages, expressions having similar meanings may (and often must) take substantially different forms. Words are drawn from discrete, coarse-grained vocabularies, hence incremental differences in meaning force discrete changes in word selection. Further, small differences in meaning are often encoded in structural choices—active *vs.* passive voice, parallel *vs.* non-parallel constructs, alternative orderings of lists, clauses, sentences, and so on. These expressive mechanisms routinely entangle semantics with structure in ways that impose incompatible graph structures on expressions that convey similar, seemingly parallel meanings.

The nature of QNR expressions invites greater regularization: Meanings of incommensurate kinds may often be encoded in substantially different graph topologies, while expressions with meanings that are similar—or quite different, yet abstractly parallel—can be represented by identical graph topologies in which differences are encoded in the continuous embedding spaces of node attributes.

Humans compare meanings after decoding language to form neural representations; natively disentangled, regularized, *alignable* representations can enable comparisons that are more direct. By facilitating alignment and comparison, regularization can facilitate systematic (and even semi-formal) reasoning.

8.4.2 Aligning Representations

Section 7.2.4 took note of neural algorithms that can align graphs in which parallel subgraphs carry corresponding attributes. Given a pair of aligned graphs, further processing can exploit these explicit correspondences.[1]

8.4.3 Alignment for Pattern-Directed Action

By triggering pattern-dependent actions, graph alignment can establish relationships between other entities that do not themselves align. For example, reasoning from facts to actions can often be formulated in terms of production rules[2] that update a representation-state (or in some contexts, cause an external action) when the pattern that defines a rule's precondition matches a pattern in a current situation: Where patterns take the form of graph-structured representations, this (attempted) matching begins with (attempted) graph alignment. Potential and actual applications of production rules range from perceptual interpretation to theorem proving.

8.4.4 Comparing, Combining and Extending Content

Alignment facilitates comparison. When graphs align, natural distance metrics include suitably weighted and scaled sums of distances between pairs of vector attributes.[3] When alignment is partial, subgraphs in one expression may be absent from subgraphs in the other (consistent with compatibility), or subgraphs may overlap and clash. Clashes may indicate mutual irrelevance or conflict, depending on semantic contents and problem contexts.

Overlapping graphs can (when compatible) be merged to construct extended, consistent descriptions of some entity or domain. In formal systems, this operation corresponds to unification (Section A1.1.1); in general QNR contexts, formal unification can be supplemented or replaced by soft, learned approximations (Section A1.4.3).

Expressions that are similar and compatible yet not identical may provide different information about a single object or abstraction; unification mechanisms then can enrich representations by merging information from

1. In this connection, consider potential generalizations of the image-domain algorithm described in Sarlin *et al.* (2020), which employs end-to-end training to recognize, represent, and align features in pairs of data objects through GNN processing followed by differentiable graph matching. Analogous processing of QNR representations would enable semantics-based alignment even in the absence of strict structural matches.

2. In recent work on neural image interpretation, Goyal *et al.* (2021) employed propositional knowledge in the form of production rules (condition-action pairs) that are represented by model parameters and selected at inference time by attention mechanisms based on similarity between condition and activation vectors.

3. Distance metrics may result from neural computations that condition on broader semantic content, or on other relationships beyond simple pairwise sums and differences.

multiple sources. For example, unification of multiple, related abstractions (*e.g.*, derived from uses of a particular mathematical construct in diverse domains of theory and application) could produce representations with richer semantics than could be derived from any single NL expression or document—representations that capture uses and relationships among abstractions, rather than merely their formal structures.

Alternatively, expressions that are similar yet not fully compatible may describe different samples from a single distribution. The dual of unification, anti-unification (a.k.a. generalization, Section A1.1.2), provides the most specific representation that is compatible with a pair of arguments, encompassing their differences (including clashes) and retaining specific information only when it is provided by both. Anti-unification of multiple samples[1] yields generalizations that can (perhaps usefully) inform priors for an underlying distribution.

Like unification, generalization yields an expression at the same semantic level as its inputs, but to represent *analogies* calls for an output that contains representations of both input graphs *and their relationships.* Alignment can provide a basis for representing analogies as first-class semantic objects that can be integrated into an evolving corpus of QNR content

---

Looking back for a moment, prospects that were motivated by the most basic considerations—upgrading words and syntax to machine-native embeddings and graphs—have naturally led to prospects that go far beyond considerations of expressiveness *per se.* QNR/NL$^+$ frameworks lend themselves to applications that can subsume yet are qualitatively different from those enabled by natural language.

1. The lattice axioms (Section A1.2) ensure that pairwise combination extends to multiple arguments in a natural way.

9 Scaling, Refining, and Extending QNR Corpora

Scalable QNR systems with NL$^+$-level expressive capacity could be used to represent, refine, and integrate both linguistic and non-linguistic content, enabling systems to compile and apply knowledge at internet scale.

Preceding sections discussed vector/graph representations and their potential use in constructing QNR frameworks that are powerful and tractable enough to meet the criteria for NL$^+$. The present section explores the potential scaling of QNR/NL$^+$ systems to large corpora, then considers how scalable, fully functional systems could be applied to include and integrate both linguistic, and non-linguistic content while refining and extending that content through judgment, unification, generalization, and reasoning.

9.1 Organizing and Exploiting Content at Scale

A core application of language-oriented QNR systems will be to translate and digest large NL corpora into tractable, accessible NL$^+$ representations, a task that makes scalability a concern. Current practice suggests that this will be practical. NL $\rightarrow$ NL$^+$ translation will likely employ components that resemble today's translation systems. Potentially analogous NL $\rightarrow$ NL translation tasks are performed at scale today, while ubiquitous internet-scale search suggests that large corpora of NL$^+$ knowledge can be exploited for practical use.

9.1.1 Learning by Reading Can Be Efficient at Scale

To estimate the magnitude of required computational tasks, we can *roughly* equate the incremental costs of building a corpus by reading NL texts and translating them to NL$^+$ representations with the incremental costs of training a language model by reading NL texts and (in some sense) "translating" their content to model parameters. According to this estimate, training GPT-3 was *roughly* equivalent to translating ~300 billion words[1] to NL$^+$ representations—about 30 times the content of the 1.8 million papers on the arXiv (arXiv 2021) or 10% of the 40 million books scanned by Google (H. Lee 2019). As an alternative and perhaps more realistic estimate, we can *roughly* equate the cost

1. Summing over products of training epochs and corpus sizes (Brown *et al.* 2020).

of NL $\rightarrow$ NL^{+} translation to the cost of NL $\rightarrow$ NL translation; the throughput of Google Translate (>3 trillion words per year[1]) is then an indicator of affordable throughput (~40 million books/month). These comparisons concur in suggesting the feasibility of translating NL information to NL^{+} corpora at scale.[2]

Translating NL content into accessible, quasicognitive NL^{+} is a form of learning that is distinct from training. Unlike training model parameters, the task of reading, translating, and expanding corpora is by nature fully parallelizable and scalable. The products are also more transparent: Exploiting external, compositional representations of information can enable facile interpretation, correction, redaction, and update of a system's content.[3]

9.1.2 Semantic Embeddings Enable Semantic Search

NL^{+} expressions can be indexed by summary embeddings that associate similar expressions with near-neighbor points in a semantic space.[4] Using these embeddings as keys in near-neighbor retrieval can provide what is in effect "associative memory" that supports not only search, but tasks involving knowledge comparison and integration (see Section 9.4 below).

Different embeddings of an expression can serve as keys suited for different search tasks. In some use-contexts, we care about the physical properties of an object; in others, we care about its cost and functionality. In some use-contexts, we care about a city's geography; in others, about its nightlife. Accordingly,

1. Reported in (Turovsky 2016).

2. Contributing to efficiency and scalability, computations that both write and read NL^{+} corpora can, as noted above, avoid problems that stem from representing knowledge solely in trained model parameters. Large language models learn not only knowledge of language *per se* and general features of the world, but also a range of specific facts about people, places, *etc.* The striking improvements that have resulted from scaling language models have stemmed, not only from improvements in broadly applicable, immediately available syntactic, semantic, and reasoning skills (K. Lu *et al.* 2021), but from memorization of specific, seldom-used facts about the world (Yian Zhang *et al.* 2020). For example, GPT-2 uses its 1.5 billion parameters to memorize names, telephone numbers, and email addresses, as well as the first 500 digits of π (Carlini *et al.* 2021). Encoding factual information of this sort in billions of model parameters—all used in computing each output step—is problematic: Both training and inference are expensive, and the results are opaque and difficult to correct or update. Parameter counts continue to grow (Brown *et al.* 2020; Fedus, Zoph, and Shazeer 2021).

3. For related work, see Verga *et al.* (2020), Lewis *et al.* (2020), Guu *et al.* (2020), and Févry *et al.* (2020).

4. Wang and Koopman (2017), Schwenk and Douze (2017), and Tran *et al.* (2020). Note that retrieval based on similarity between semantic embeddings can be effective across modalities (*e.g.*, Miech *et al.* 2021). Pairwise similarity between vector keys could potentially reflect the unification scores (Section A1.4.3) of the corresponding pairs of expressions.

what might be represented as a single span of content may naturally be associated with multiple domain-oriented summaries and keys. We find this general pattern in Transformers, where multi-head attention layers project each value to multiple key and query vectors. A quite general distinction is between representing the *content* of an expression and representing the *kinds of questions* that its content can answer—between what it *says* and what it is *about*.

The role of semantic search overlaps with the role of links in large-scale NL^+ structures (for example, generalizations of citation networks), but differs in designating *regions of semantic space* through query-embeddings rather than designating *specific expressions* through graph links. Conventional references and syntactic relationships are represented by links, but looser relationships can be represented by "query embeddings" *within expressions*, where these embeddings are taken to denote soft graph links to a potential multiplicity of target expressions in an indexed corpus.[1]

9.1.3 Semantic Search Extends to Non-Linguistic Content

By proposed construction, any item that can be described by NL can be (better) described by NL^+. Examples include code, engineering designs, legal documents, biomedical datasets, and the targets of current recommender systems—images, video, apps, people, products, and so on. Embeddings that represent (descriptions of) non-linguistic content (Section 9.2.2) are relevant to NL^+ in part because they can be directly referenced by NL^+ expressions, and in part because these NL^+ descriptions can provide a basis for semantically rich content embeddings.

9.1.4 NL^+-Mediated Embeddings Could Potentially Improve NL Search

Systems trained to map NL to NL^+ can support the production of NL embeddings ($NL \rightarrow NL^+ \rightarrow$ embedding) that are based on disentangled, contextually informed semantic representations. If so, then co-learned, NL^+-mediated key and query embeddings could potentially improve *k*NN semantic search over corresponding NL items at internet scale.

1. Note that this mechanism provides a *differentiable* and potentially fluid graph representation; see Section 7.2.3.

9.1.5 Semantic Search Can Be Efficient at Scale

Exact k-nearest neighbor search in generic high-dimensional spaces is costly,[1] but approximate nearest neighbor search (which usually finds the keys nearest to a query) have been heavily researched and optimized; efficient, practical, polylogarithmic-time algorithms can support (for example) billion-scale recommender systems,[2] and can do likewise for retrieval of semantic content in NL$^+$ systems at scale.

9.2 Incorporating General, Non-Linguistic Content

Human knowledge includes extensive nonlinguistic content, yet we use natural language to describe, discuss, index, and provide instructions on how to create and use that content. Language-linked nonlinguistic content sprawls beyond the boundaries of an NL-centric concept of NL$^+$—boundaries that do not constrain language-inspired QNR/NL$^+$ applications.

Examples of non-linguistic information content include:

- Non-linguistic lexical-level units (*e.g.*, image embeddings)
- Expressions in formal languages (*e.g.*, mathematical proofs)
- Precise descriptions of objects (*e.g.*, hardware designs)
- Formal graph-structured representations (*e.g.*, probabilistic models)

9.2.1 Using "Words" Beyond the Scope of Natural Language

"Nouns" represent things, but vector embeddings can represent things in ways that are in no sense translations of word-like entities in NL: Image embeddings, for example, can serve as "nouns",[3] though the content of an image embedding need not resemble that of an NL noun phrase. Embeddings that represent datasets or object geometries have a similar status.

Similar considerations apply to verbs and relationships: Words and phrases have a limited capacity to describe motions, transformations, similarities,

1. Scaling as ~$O(n)$, but learned, structured key spaces can improve scaling to $O(n^{1/2})$ (Lample *et al.* 2019).

2. J. Wang *et al.* (2018), Fu *et al.* (2018), Johnson, Douze, and Jégou (2019), Jayaram Subramanya *et al.* (2019), and Sun (2020)

3. Note that representation of images through relationship-graphs among objects blurs the boundary between opaque image embeddings and syntactic structures; See for example Bear *et al.* (2020). Image-embedding spaces can also be aligned with text (*e.g.*, in Patashnik *et al.* 2021).

and differences.[1] As with nouns, embeddings (and a wide range of other information objects) can again subsume aspects of NL expressive capacity and extend representations to roles beyond the scope of practical NL descriptions. This expressive scope may often be difficult to describe concretely in NL.[2] QNR "verbs" could express transformations of kinds not compactly expressible in conventional NL: For example, displacements of physical objects can be represented by matrices that quantitatively describe displacements and rotations, and a more general range of transformations can be expressed by displacement vectors in a suitable latent space. Stretching the conceptual framework further, verb-like transformations can be specified by executable functions.[3]

9.2.2 Referencing Non-Linguistic Objects

References to linguistic and non-linguistic objects are not sharply demarcated, but some information objects are both complex and opaque, while physical objects are entirely outside the information domain. All these (indeed, essentially anything) can nonetheless be referenced within the syntactic frameworks of QNR expressions.

As a motivating case, consider how embedded hyperlinks and more general URIs expand the expressive power of online documents: The ability to unambiguously reference not only text, but arbitrary information objects, is powerful, and on the internet, this capability meshes smoothly with NL.

Examples of non-linguistic information objects include images, websites, data repositories, and software. Beyond the domain of information objects, domain-appropriate reference modalities can act as proper nouns in designating physical objects, people, places, and the like. A natural pattern of use would place a reference in a QNR wrapper that might include (for example) a summary embedding, conventional metadata, descriptions, and documentation, all of which can exploit the representational capacity of NL$^+$. QNR wrappers can facilitate indexing, curation, and the selection or rejection of entities for particular uses.

1. For example, an NL phrase can compactly say that "face_1 strongly resembles face_2", while a lexical-level embedding can compactly say that: "face_1, with a specific set of embedding-space offsets in shape, color, and expression, looks like face_2 with some quantified but approximate residual differences."

2. Hence the value of using interpretable yet non-linguistic image embeddings as examples.

3. In this connection, note that QNR expressions are data, that neural functions can be of type QNR $\rightarrow$ QNR, and that "apply", "eval" and "quote" functions can play quite general roles (*e.g.*, in languages like Scheme that can express and operationalize high-level abstractions).

Information objects include executable code, and when accessed remotely, executable code is continuous with general software and hardware services. Interactive models[1] can complement NL descriptions of systems. Access to documented instances of the computational models used to produce particular NL^+ items can contribute to interpreting the items themselves: Connections between products and their sources often should be explicit.

9.2.3 Embedding Graph-Structured Content

Human authors routinely augment sequential NL text with graph structures. Even in basically sequential text (*e.g.*, this document), we find structures that express deep, explicit nesting and internal references. Documents often include diagrams that link text-labeled elements with networks of arrows, or tables that organize text-labeled elements in grids. These diagrams and tables correspond to graphs.

Further afield, graph-structured representations can describe component assemblies, transportation systems, and biological networks (metabolic, regulatory, genetic, *etc.*) Text-like descriptions of statistical and causal relationships become probabilistic models and causal influence diagrams. In ML, we find formal knowledge graphs in which both elements and relationships are represented by vector embeddings,[2] while augmenting language-oriented Transformer-tasks with explicit representations of relationships has proved fruitful.[3] Distinctions between these and NL^+ representations blur or disappear when we consider generalizations of conventional syntax to graphs, and of symbols and text to embeddings and general data objects. Some potential use-patterns can be conceptualized as comprising restricted, graph-structured representations (*e.g.*, formal structures that support crisply defined inference algorithms), intertwined with fully general graph-structured representations (informal structures that support soft inference, analogy, explication of application scope, and so on).

9.2.4 Integrating Formal Quasilinguistic Systems

Programming languages and mathematics are formal systems, typically based on quasilinguistic representations.[4] In these systems, the word-like entities

1. *E.g.*, the models presented in Distill pages (Distill Team 2021).
2. Kazemi and Poole (2018) and Hogan *et al.* (2021)
3. Currey and Heafield (2019), Schlag *et al.* (2020), and Nguyen *et al.* (2020)
4. In a QNR context, formal graph representations (such as diagrams in category theory) can be regarded as languages that generalize text-based syntax.

(symbols) have minimal semantic content, yet syntactic structures in conjunction with an interpretive context specify semantic or operational meanings that endow these systems with descriptive capabilities beyond the conventional scope of NL.

How are formal systems connected to NL, and by extension, to NL$^+$ frameworks? NL cannot replace formal systems, and experience suggests that no conventional formal system can replace NL. What we find in the wild are formal structures combined with linguistic descriptions: mathematics interleaved with explanatory text in papers and textbooks, programs interleaved with comments and documentation in source code, and so on. Experience with deciphering such documents suggests the value of intimate connections between NL-like descriptions and embedded formal expressions.[1] In one natural pattern of use, formal systems (*e.g.*, mathematics, code, and knowledge representation languages) would correspond to distinguished, formally interpretable subsets of networks of NL$^+$ expressions.

Formal languages can describe executable operations to be applied in the informal context of neural computation. Conversely, vector embeddings can be used to guide premise selection in the formal context of theorem proving.[2]

9.3 Translating and Explaining Across Linguistic Interfaces

Proposed NL$^+$ content has both similarities to NL and profound differences. It is natural to consider how NL might be translated into NL$^+$ representations, how NL$^+$ representations might be translated or explained in NL, and how anticipated differences among NL$^+$ dialects might be bridged.

9.3.1 Interpreting Natural Language Inputs

NL$^+$ frameworks are intended to support systems that learn through interaction with the world, but first and foremost, are intended to support learning from existing NL corpora and language-mediated interactions with humans. Translation from NL sources to NL$^+$ is central both to representation learning and to key applications.

It is natural to expect that encoders for NL $\rightarrow$ NL$^+$ translation will share a range of architectural characteristics and training methods with encoders for NL $\rightarrow$ NL translation and other NLP tasks (Section 10.2). Translation to NL$^+$

1. Szegedy (2020) suggests an NL-translation approach to formalizing mathematics papers.
2. M. Wang *et al.* (2017) and Minervini *et al.* (2018)

could be applied both to support immediate tasks and (more important) to expand corpora of NL$^+$-encoded knowledge.

Transformer-based NL $\rightarrow$ NL translation systems can learn *language-agnostic representations,*[1] a capability which suggests that NL $\rightarrow$ NL$^+$ translation will be tractable.

9.3.2 Translating Among NL$^+$ Dialects

Because NL$^+$ corpora could span many domains of knowledge—and be encoded by multiple, independently trained systems—it would be surprising to find (and perhaps challenging to develop) universally compatible NL$^+$ representations. In neural ML, different models learn different embeddings, and the representations learned by models with different training sets, training objectives, and latent spaces may diverge widely. In an NL$^+$ world, we should expect to find a range of NL$^+$ "dialects" as well as domain-specific languages.

Nonetheless, where domain content is shared and representational capacities are equivalent, is reasonable to expect facile NL$^+$ $\rightarrow$ NL$^+$ translation. Further, regarding interoperability in non-linguistic tasks, the concrete details of differing representations can be hidden from clients by what is, in effect, an abstraction barrier.[2] Domain-specific languages may resist translation at the level of representations, yet contribute seamlessly to shared, cross-domain functionality.

Lossless translation is possible when the semantic capacity of one representation fully subsumes the capacity of another. Given the computational tractability of NL$^+$ representations, we can expect translation between similar NL$^+$ dialects to be more accurate than translation between natural languages. In translation, spaces of lexical-level embeddings can be more tractable than discrete vocabularies in part because vector-space transformations can be smooth and one-to-one.[3]

9.3.3 Translating and Explaining NL$^+$ Content in NL

It is reasonable to expect that, for a range of NLP tasks, conventional NL $\rightarrow$ (opaque-encoding) $\rightarrow$ NL pipelines can be outperformed by NL $\rightarrow$ NL$^+$ $\rightarrow$

1. Y. Liu *et al.* (2020), Tran *et al.* (2020), and Botha, Shan, and Gillick (2020)

2. Object-oriented programming exploits this principle.

3. Even projections of sets of high-dimensional vectors into spaces of substantially lower dimensionality can preserve key geometric relationships quite well: The Johnson–Lindenstrauss lemma implies good preservation of both distances and cosine similarities between vectors.

NL pipelines;[1] this would imply effective translation of the intermediate NL$^+$ representations to NL.

If NL$^+$ frameworks fulfill their potential, however, NL$^+$ corpora will contain more than translations of NL expressions. Content will typically draw on information from multiple sources, refined through composition and inference, and enriched with non-linguistic word-like elements. There is no reason to expect that the resulting representations—which will not correspond to particular NL expressions or any NL vocabularies—could be well-translated into NL. If NL$^+$ is more expressive than NL, it follows that not all NL$^+$ content can be expressed in NL.

How, then, might NL$^+$ content be accessed through NL, either in general or for specific human uses?

- Some NL$^+$ expressions will correspond closely to NL expressions; here, we can expect to see systems like conditional language models (Keskar *et al.* 2019) applied to produce fluent NL translations.
- NL$^+$ descriptions that are detailed, nuanced, complex, and effectively untranslatable can inform NL descriptions that provide contextually relevant information suitable for a particular human application.
- Similarly, a general abstraction expressed in NL$^+$ might be effectively untranslatable, yet inform narrower, more concrete NL descriptions of contextually relevant aspects of that abstraction.
- To the extent that NL expressions could—if sufficiently extensive—convey fully general information (a strong condition), NL could be used to describe and explain NL$^+$ content in arbitrary detail; this approach is continuous with verbose translations.
- NL$^+$ content in effectively non-linguistic domains could in some instances be expressed in diagrams, videos, interactive models, or other human-interpretable modalities.

Relative to the opaque representations common in current neural ML, NL$^+$ representations have a fundamental advantage in interpretability: Because QNRs are compositional, their components can be separated, examined, and perhaps interpreted piece by piece. Even when components cannot be fully interpreted, they will often refer to some familiar aspect of the world, and knowing *what an expression is about* is itself informative.

1. Particularly when intermediate NL$^+$ processing can draw on relevant NL$^+$ corpora. Successful augmentation of Transformers with external memory (*e.g.*, for question answering) provides evidence for the potential power of this approach (Koncel-Kedziorski *et al.* 2019; Fan *et al.* 2021; Min *et al.* 2020; Thorne *et al.* 2020).

9.4 Integrating and Extending Knowledge

Graph-structured representations in which some vector attributes designate regions in semantic spaces[1] lend themselves to operations that can be interpreted as continuous relaxations of formal unification and anti-unification, which in turn can support reasoning and logical inference by (continuous relaxations of) familiar algorithms. These operations can help extend corpora by combining existing representations along lines discussed from a somewhat different perspective in the preceding section.

9.4.1 Combining Knowledge Through (Soft) Unification

Compatible representations need not be identical: Alignment and (successful) soft unification (Appendix A1) indicate compatibility, and the successful unification of two expressions defines a new expression that may both combine and extend their graph structures and semantically narrow their attributes. Soft unification could potentially be used to refine, extend, compare, and link QNR/NL$^+$ representations. Where QNR graphs partially overlap, successful unification yields a consistent, extended description.[2] Attempts to unify incompatible representations fail and could potentially yield a value that describes their semantic inconsistencies.[3]

In refining content through soft unification, relatively unspecific structures in one QNR[4] may be replaced or extended by relatively specific structures[5]

1. A simple example would be regions implied by implicit, contextual uncertainty in a vector value; richer, more formal examples include spaces in which vector values (points) explicitly correspond to regions in lower-dimensional spaces, or in which points are semantically related by taxonomic or set-inclusion relationships. In a limiting case, vector values correspond either to points (which, through comparison by equality, can model mathematical symbols) or to unknowns (which, through co-reference, can model mathematical variables).

2. As a non-linguistic analogy, consider overlapping fragments of an image: Where overlaps match well enough, the fragments can be glued together to form a more comprehensive image of a scene, combining information from both fragments and potentially revealing new relationships.

3. Section 10.6.7 suggests generically applicable training objectives that would favor representations and operations that (approximately) satisfy the axioms (Section A1.2) for unification and anti-unification; in this approach, operations may be performed by *contextually informed* neural functions.

4. *E.g.*, embeddings that represent uncertain values; nodes that lack links to optional content; leaf-level nodes that in effect summarize the content of some range of potential graph extensions.

5. *E.g.*, embeddings that represent narrower values; links to graph structure that may be modified by conditioning on a compatible leaf-level embedding.

from an aligned QNR. Embeddings that contain *different* information may commbine to yield a semantically narrower embedding.

Products of successful unification (new, more informative expressions and relationships) are candidates for addition to an NL$^+$ corpus. Records of failed unifications—documenting specific clashes—can provide information important to epistemic judgment. These successful and unsuccessful unification-products may correspond to nodes in a semantically higher-level graph that represents relationships among expressions.

9.4.2 Generalizing Knowledge Through (Soft) Anti-Unification

While unification of two expressions can be regarded as combining their information content, anti-unification (generalization) can be regarded as combining their uncertainties, spanning their differences, and discarding unshared information. Generalization in this sense may represent a useful prior for generative processes within distributions that include the inputs.

9.4.3 Constructing Analogies

Operations based on generalization and unification could be applied to identify, construct, and apply analogies and abstractions in QNR corpora:

- A range of latent analogies will be reflected in recognizably parallel structures between or among concrete descriptions.[1]
- Alignment of parallel concrete descriptions can establish concrete analogies, potentially reified as graphs.
- Generalization over sets of parallel descriptions can abstract their common structures as a patterns.
- Unification of a concrete description with a pattern can indicate its analogy to a set of similar concrete descriptions without requiring pairwise comparison.

Analogies have powerful applications. For example, if a description of A includes features of a kind absent from the analogous description of B, then it is reasonable to propose A-like features in B. Analogies among mammals, for example, underlie the biomedical value of discovery in animal models.

1. The scope of recognizable parallels will depend on learned representations and comparison operators. Regularization (Section 8.4) can make representations more comparable; useful comparison operators could potentially resemble relaxed unification and generalization operators.

Indeed, analogy permeates science, guiding both hypotheses and research planning.

Analogy has been identified as central to cognition,[1] and reified networks of analogies can form graphs in which relationships among abstractions are themselves a domain of discourse. With suitable annotations, products of generalization and analogy—new abstract expressions and relationships—are candidates for addition to an NL$^+$ corpus.

9.4.4 Extending Knowledge Through (Soft) Inference

Natural language inference (NLI) is a major goal in NLP research, and recent work describes a system (based on a large language model) in which NL statements of rules and facts enable answers to NL questions (Clark, Tafjord, and Richardson 2020). Inference mechanisms that exploit NL$^+$ $\rightarrow$ NL$^+$ operations could potentially be useful in NLI pipelines, and in refining and extending NL$^+$ corpora NL$^+$ $\rightarrow$ NL$^+$ inference could play a central role.

Regularizing and normalizing QNR representations (Section 8.4) can enable a kind of "soft formalization" based on continuous relaxations of formal reasoning (modeled, for example, on logic programming, Section A1.4.1). Rules can be represented as "if-then" templates (in logic, expressions with unbound variables) in which successful unification of an expression with an "if-condition" template narrows the values of attributes that, through coreference, then inform expressions constructed from "then-result" templates.[2]

Advances in neural mechanisms for conventional symbolic theorem-proving (*e.g.*, guiding premise selection) have been substantial.[3] It is reasonable to expect that wrapping formal expressions in NL$^+$ descriptions—including embeddings and generalizations of potential use contexts—could facilitate heuristic search in automated theorem proving.

9.5 Credibility, Consensus, and Consilience

Humans examine, compare, contrast, correct, and extend information represented in NL literatures. Machines can do likewise with NL$^+$ content, and for

1. See Hofstadter (2009) and Gentner and Forbus (2011).

2. Unification-based machinery of this kind can implement Prolog-like computation with applications to natural language inference (Weber *et al.* 2019).

3. Rocktäschel and Riedel (2016, 2017), Minervini *et al.* (2018), and Minervini *et al.* (2019)

similar purposes,[1] a process that can exploit unification (and failure), together with generalization and analogy.

9.5.1 Modeling and Extending Scholarly Literatures

The strategy of using NL as a baseline suggests seeking models for NL$^+$ corpora in the scholarly literature, a body of content that includes both structures that are broadly hierarchical (*e.g.*, summary/body and section/subsection relationships) and structures that correspond to more general directed graphs (*e.g.*, citation networks).

In abstracts, review articles, and textbooks, scholarly literatures summarize content at scales that range from papers to fields. Proposed NL$^+$ constructs can support similar patterns of expression, and can extend content summarization to finer granularities without cluttering typography or human minds.

Scholarly citations can link to information that is parallel, supportive, problematic, explanatory, or more detailed; in NL$^+$ syntax, analogous citation functionality can be embodied in graphs and link contexts.

Through indexing, citations in scholarly literatures can be made bidirectional,[2] enabling citation graphs to be explored through both cites-x and cited-by-x relationships. For similar reasons, NL$^+$ links in fully functional systems should (sometimes) be bi-directional.[3] In general, the structure and semantics of citations and citing contexts can vary widely (document-level remote citations are continuous with sentence-level coreference), and the structure of NL$^+$ representations makes it natural to extend *cites* and *cited-by* relationships to expressions finer-grained than NL publications.

Steps have been taken toward applying deep learning to improve the integration of scholarly literatures.[4] It will be natural to build on this work using NL$^+$ tools to enrich NL$^+$ content .

9.5.2 Using Credibility, Consensus, and Consilience to Inform Judgments

As with NL, not all NL$^+$ content will be equally trustworthy and accurate. The origin of information—its provenance—provides evidence useful in judging

1. Argumentation mining points in this direction (Moens 2018; Galassi *et al.* 2020; Slonim *et al.* 2021).

2. In an awkward coarse-grained manner.

3. Note, however, that cited-by relationships can have massive fanout, a pattern of use that may call for backward-facing structures richer than link-sets.

4. Jaradeh *et al.* (2019), M. Jiang *et al.* (2020), and Cohan *et al.* (2020)

epistemic quality, a consideration that becomes obvious when considering information derived from heterogeneous NL sources. Judgments of epistemic quality can reflect not only judgments of individual sources (their *credibility*), but also the consistency of information from different sources, considering both *consensus* among sources of similar kind, and *consilience* among sources that differ in kind.

Search by semantic similarity and comparison through structural and semantic alignment provide a starting point, but where epistemic quality is in question, provenance will often be key to resolving disagreements. Some broad distinctions are important.

9.5.2.1 Informing judgments through provenance and credibility

Provenance is an aspect of context that calls for summarization. In a fully functional system, embeddings (and potentially extended QNRs[1]) can summarize both information sources and subsequent processing, providing descriptive information that can be linked and accessed to derived content for deeper examination. Provenance information helps distinguish broad concurrence from mere repetitions—without tracking sources, repetitions may wrongly be counted as indicating an informative consensus.

By analogy with (and leveraging) human judgment, systems can with some reliability recognize problematic content that can range from common misconceptions through conspiracy theories, fake news, and computational propaganda.[2] Problematic content should be given little weight as a source of information about its subject, yet may itself be an object of study.[3] Judgments of quality can be iterative: The quality and coherence of content can be judged in part by the quality and coherence of its sources, and so on, a process that may converge on descriptions of more-or-less coherent but incompatible models of the world together with accounts of their clashes.

Judging source quality in generally sound NL literatures is a familiar human task. In the experimental sciences, for example, we find a spectrum of epistemic status that runs along these lines:

- Uncontroversial textbook content
- Reviews of well-corroborated results

1. Summaries (like other expressions) can be embodied in QNRs that provide increasing detail with increasing syntactic depth.

2. See Martino *et al.* (2020).

3. In this context, the difference between toxic text and discussions that embed examples of toxic text illustrates the importance of recognizing use-mention distinctions. Social media filters today may suppress both advocates and critics of offensive views.

- Reports of recent theory-congruent results
- Reports of recent surprising results
- Reports of recent theory-incongruent results
- Reports of actively disputed results
- Reports of subsequently retracted results

All of these considerations are modulated by the reputations of publishers and authors.

Broadly similar indicators of quality can be found in history, economics, current affairs, and military intelligence. The reliability of sources is typically domain-dependent:[1] Nobel laureates may speak with some authority in their fields, yet be disruptive sources of misinformation beyond it; a conspiracy theorist may be a reliable source of information regarding software or restaurants. Although information about credibility can be propagated through a graph, credibility is not well-represented as a scalar.

9.5.2.2 Informing judgments through consensus

In judging information, we often seek multiple sources and look for areas of agreement or conflict—in other words, degrees of consensus. Relevant aspects of provenance include the quality of individual sources, but also their diversity and evidence of their independence.[2] What amount to copying errors may be indicated by sporadic, conflicting details. Lack of independence can often be recognized by close similarity in how ideas are expressed.[3]

In judging information from (what are credibly considered to be) direct observations, experiments, and experience, the quality of human sources may play only a limited role. Established methods of data aggregation and statistical analysis will sometimes be appropriate, and while NL$^+$ representations may be useful in curating that data, subsequent methods of inference may have little relationship to NL$^+$ affordances. Inference processes themselves, however, constitute a kind of algorithmic provenance relevant to downstream representation and assessment of results.

9.5.2.3 Informing judgments through consilience

More powerful than consensus among sources of broadly similar kinds is *con-*

1. Domain-based assessments of credibility have been used in constructing knowledge graphs from social-media sources: Abu-Salih *et al.* (2021).

2. Some of this evidence is found in surface features of NL texts (*e.g.*, uses of specific words and phrases); other evidence is found in features of semantic content.

3. Here, links to NL source text can be valuable: Literal wording may convey signals of shallow repetition.

silience, agreement of evidence from sources of qualitatively different kinds—for example, agreement between historical records and radiocarbon dating, or between an experimental result and a theoretical calculation. Judgment of what constitutes "a difference in kind" is a high-level semantic operation, but potentially accessible to systems that can recognize similarities and differences among fields through their citation structures, focal concerns, methodologies, and so on. Distinguishing consilience from mere consensus is a judgment informed in part by provenance, and is key to robust world modeling. It calls for modeling the epistemic structures of diverse areas of knowledge.

10 Architectures and Training

Extensions of current neural ML methods can leverage architectural inductive bias and multitask learning to support the training of quasilinguistic neural systems with NL$^+$-level expressive capacity.

The preceding sections suggest that QNR frameworks can implement powerful, tractable NL$^+$ functionality, provided that suitable representations can be learned; the present section outlines potentially effective approaches to learning based on adaptations of familiar architectures and training methods. Vector-labeled graph bottlenecks can provide a strong inductive bias, while multitask learning and auxiliary loss functions can shape abstract representations that are anchored in, yet substantially decoupled from, natural languages. The final section outlines potential architectures for components that control inference strategies.

10.1 General Mechanisms and Approaches

Following neural ML practice, the development of QNR-centered systems calls, not for hand-crafting features, but for architectures that provide suitable components, capacity, and inductive bias, in conjunction with training tasks that provide suitable datasets, objectives, and loss functions. General mechanisms and approaches include:

- Employing NL $\rightarrow$ QNR interfaces to ensure QNR representations
- Employing QNR intermediate representations in NL $\rightarrow$ NL training tasks
- Decoupling QNR representations from NL encodings
- Employing semantically rich training tasks with QNR objectives

- Structuring QNR semantics through auxiliary, lattice-oriented training
- Applying QNR-domain inference to exploit QNR repositories

These mechanisms and approaches should be seen as facets of multitask learning in which a key goal is to develop NL → QNR → NL systems[1] that support broad applications. Pretrained NL → embedding → NL models (*e.g.*, BERT and friends) are perhaps today's closest analogues.

10.2 Basic Information Flows

Developing QNR-centered systems calls for encoders and decoders that can link input and output channels to QNR-based representation and inference mechanisms (Figure 10.1).

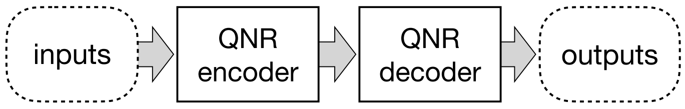

Figure 10.1: *Information flows in minimalistic, QNR-bottleneck systems. Inputs and outputs may be multimodal.*

- Potential *inputs* to encoders include text, but also images, symbolic expressions, multimodal data streams,[2] and so forth.
- Potential *outputs* from decoders include translations, summaries, answers to questions, retrieved content, classifications, and various products of downstream processing (images, engineering designs, agent behaviors, and so on).
- Potential *QNR operations* range from simple pass-through (implementing a QNR bottleneck[3] without QNR-domain inference) to inference mechanisms that could employ QNR-based unification, generalization, and reasoning methods[4] while drawing on stored QNR content (Figure 10.2).

1. Together with generalizations to multimodal inputs and outputs.

2. See for example J. Lu *et al.* (2019) and Desai and Johnson (2021).

3. It is at least questionable whether the inductive bias provided by a simple QNR-bottleneck architecture would outperform an otherwise similar but unconstrained encoder/decoder architectures in stand-alone NL → NL tasks. The use of a QNR-like bottleneck in Bear *et al.* (2020) has led to strong performance, but small-scale QNR representations can be interchangeable with vector embeddings that lack explicit graph structure (Section 7.3.1).

4. Note that open-ended reasoning likely calls for conditional computation; potentially relevant architectural components and training methods are discussed in Cases *et al.* (2019), Rosenbaum *et al.* (2019), and Banino, Balaguer, and Blundell (2021).

Potential architectural building blocks range from MLPs and convolutional networks to Transformers and the many varieties of graph neural networks.[1] Architectures can employ building blocks of multiple kinds that collectively enable differentiable end-to-end training (Section 7.2.3 discusses differentiable representations of graph topology).

The current state of the art suggests Transformer-based building blocks as a natural choice for encoding NL inputs and generating NL outputs. Transformer-based models have performed well in knowledge-graph → text tasks, (Ribeiro *et al.* 2020), and can in some instances benefit from training with explicit syntax-graph representations.[2]

Encoders like those developed for scene-graph representation learning[3] are natural candidates for QNR-mediated vision tasks. In both NL and vision domains, encoders can produce vector/graph representations that, through training and architectural bias, serve as QNRs. GNNs or graph-oriented Transformers are natural choices both for implementing complex operations and for interfacing to task-oriented decoders. Simple feed-forward networks are natural choices for transforming and combining the vector components of vector-labeled graphs. Systems that read and write expressions in QNR corpora could employ scalable near-neighbor lookup in repositories indexed by QNR-derived semantic embeddings (Section 9.1.2).

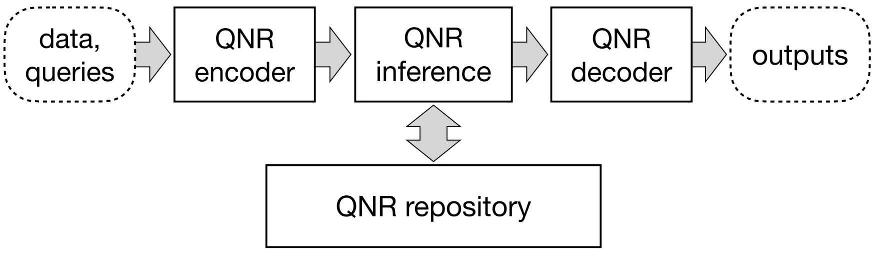

Figure 10.2: *Information flows in generic QNR systems augmented by access to a repository of QNR content. In the general case, "QNR inference" includes read/write access to repositories, producing models that are in part pre-trained, but also pre-informed.*

1. Reviewed in J. Zhou *et al.* (2020) and Wu *et al.* (2021). Transformers in effect operate on fully connected graphs, but on sparse graphs, GNNs can provide greater scalability and task-oriented inductive biases (Addanki *et al.* 2021), as well as more direct compatibility with QNRs.

2. Currey and Heafield (2019) and Akoury, Krishna, and Iyyer (2019)

3. Including both image and joint image-language models; see Zellers *et al.* (2018), J. Yang *et al.* (2018), Lee *et al.* (2019), and Bear *et al.* (2020).

The analysis presented in previous sections suggests that QNRs can in principle meet the criteria for representing NL$^+$-level semantics, while the capabilities of current neural systems suggest that architectures based on compositions of familiar building blocks can implement the operations required for NL$^+$-mediated functionality. The next question is how to *train* such systems—how to combine tasks and inductive bias to produce encoders, decoders, and processing mechanisms that provide the intended functionality.

10.3 Shaping QNR Semantics

Basic aspects of intended QNR semantics include non-trivial syntax in conjunction with NL-like representational capacity and extensions to other modalities. These goals can be pursued through architectural inductive bias together with training tasks in familiar domains.

10.3.1 Architectural Bias Toward Quasilinguistic Representations

Architectural inductive bias can promote the use of *syntactically nontrivial* QNRs (rather than flat sequences of embeddings) to represent broadly NL-like content. If learned representations follow the general pattern anticipated in Section 8, QNR syntax would typically employ (at least) DAGs of substantial depth (Section 8.1); leaf attributes would typically encode (at least) lexical-level semantic information (Section 8.2), while attributes associated with internal nodes would typically encode relationships, summaries, or modifiers applicable to subsidiary expressions (Section 8.3).

Encoders and decoders could bias QNRs toward topologies appropriate for expressing quasilinguistic semantics. Architectures can pass information through QNR-processing mechanisms with further inductive biases—*e.g.*, architected and trained to support soft unification—to further promote the expression of computationally tractable, disentangled, quasilinguistic content.

10.3.2 Anchoring QNRs Semantic Content in NL

Although QNR representations have broader applications, it is natural to focus on tasks closely tied to language. Transformers trained on familiar NL → NL objectives (*e.g.*, language modeling and sentence autoencoding) have produced flat vector representations (vectors and vector-sequences) that support an extraordinary range of tasks.[1] Training QNR-bottleneck architectures (NL →

1. X. Liu *et al.* (2019), Brown *et al.* (2020), and Y. Liu *et al.* (2020)

QNR $\rightarrow$ NL) on the same NL $\rightarrow$ NL objectives should produce comparable (and potentially superior) QNR representations and task performance. Potential tasks include:

- Autoencoding NL text
- NL language modeling
- Multilingual translation
- Multi-sentence reading comprehension
- Multi-scale cloze and masked language tasks[1]

It seems likely that developing NL$^+$-level representations and mechanisms would best be served, not by a pretraining/fine-tuning approach, but by concurrent multitask learning. In this approach, optimization for individual tasks is not an end in itself, but a means to enrich gradient signals and learned representations.[2]

10.3.3 Extending QNR Representations Beyond Linguistic Domains

A further class of tasks, $X \rightarrow$ QNR$\rightarrow$ NL, would map non-linguistic inputs X to NL, again mediated by (and training) QNR-based mechanisms. Potential examples include:

- Predicting descriptions of images
- Predicting descriptions of human actions
- Predicting comments in code

A quite general class of tasks would encode information from a domain, decode to a potentially different domain, and train QNR $\rightarrow$ QNR components to perform intermediate reasoning steps. Potential examples include the control of agent behavior involving instruction, communication, and planning.[3]

10.4 Abstracting QNR Representations from NL

If NL$^+$ is to be more than a representation of NL, the training of QNR models may require an inductive bias toward representations that are deliberately decoupled from NL. Lexical-level vector embeddings already provide a useful

1. Re. multi-scale masking, see Joshi *et al.* (2020).

2. See McCann *et al.* (2018), X. Liu *et al.* (2019), Alex Ratner *et al.* (2018), and Alexander Ratner *et al.* (2020).

3. Analogous language-infused mechanisms are described in Shah *et al.* (2018), Luketina *et al.* (2019), and Lazaridou and Baroni (2020).

bias in that they decouple representations from the peculiarities of NL vocabularies. Massively multilingual tasks (translation, *etc.*) can further encourage the emergence of representations that abstract from the features of particular languages.[1] In current practice, combinations of multitask learning and architectural bias have been employed to separate higher-level and lower-level semantic representations.[2] It may be useful, however, to seek additional mechanisms for learning representations that are abstracted from NL.[3] Supporting this idea, recent work has found that disentangling semantics from NL syntax is practical and can provide advantages in performing a range of downstream tasks (Huang, Huang, and Chang 2021).

10.4.1 Abstracting QNR Representations From Word Sequences

Tasks and architectures can be structured to favor separation of abstract from word-level representations.[4] A general approach would be to split and recombine information paths in NL $\rightarrow$ NL tasks: An abstract QNR path could be trained to represent predominantly high-level semantics and reasoning, while an auxiliary path carries lexical-level information. To recombine these paths, the high-level semantic path could feed a decoder that is also provided with a set of words from the target expression permuted together with decoys.[5] By reducing the task of producing correct NL outputs to one of selecting and arranging elements from a given set of words, this mechanism could shift a lexical-level, NL-specific burden—and perhaps the associated low-level semantic content—away from the abstract, high-level path. To strengthen separation, the gradient-reversal trick for domain adaptation[6] could be applied to actively "anti-train" the availability of word-specific information in abstract-path representations.

1. See, for example, Arivazhagan *et al.* (2019), Y. Liu *et al.* (2020), and Tran *et al.* (2020).

2. Sanh, Wolf, and Ruder (2018) and Tamkin, Jurafsky, and Goodman (2020)

3. Fine-tuning to reintroduce NL-specific information would likely be useful for some NL $\rightarrow$ NL applications.

4. For example, Wieting, Neubig, and Berg-Kirkpatrick (2020) separates semantic information from language-specific information in a dual-language sentence-embedding task. See also Bousmalis *et al.* (2016).

5. Alternatively, a deep, high-level path could be trained to refine a distribution over words provided by a shallow, pretrained, high-perplexity language model.

6. Ganin and Lempitsky (2015) and Cai *et al.* (2019)

10.4.2 Strategies for Learning Higher-Level Abstractions

Objective functions in NLP often score outputs by their correspondence to specific sequences of target words. This objective is embedded in the definitions of language modeling, masked language modeling, and typical cloze tasks, while similar objectives are standard in NL translation. However, as the size of target outputs increases—from single-word cloze tasks to filling gaps on the scale of sentences, paragraphs, and beyond—predicting specific word sequences becomes increasingly difficult or effectively impossible. When the actual research objective is to manipulate representations of *meaning*, lexical-level NL training objectives fail the test of scalability.

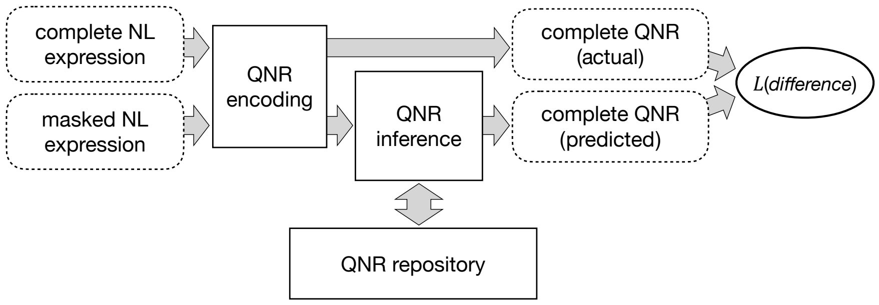

Figure 10.3: *A semantic-completion task in which loss is based on correspondence between QNR representations rather than decoded text. QNR-completion objectives can provide semantic, NL-based completion tasks, e.g., describing (not replicating) the missing components of an explanation, argument, story, proof, program, or neural architecture. To avoid collapsing representations, QNR encoders could be frozen or concurrently shaped by additional training tasks (e.g., QNR-mediated NL autoencoding, translation, etc.; see Section 10.3.2)*

Completion tasks[1] formulated *in the QNR domain itself* would better serve this purpose. Useful QNR-domain completion tasks require QNR targets that represent rich task-domain semantics, but we have already seen how NLP tasks can be used for this purpose (Section 10.3.2). Products of such training can include NL → QNR encoders that raise both inference processes *and their targets* to the QNR domain (Figure 10.3).

1. In a general sense, completion tasks can include not only sequence prediction and cloze tasks, but also question answering and other capabilities shown by language models in response to prompts (see Brown *et al.* (2020)); prediction in abstracted (latent space) domains can also support a range of tasks. See Oord, Li, and Vinyals (2019).

With targets raised from NL to QNR representations, it should become practical to compare outputs to targets even when the targets represent complex semantic objects with an enormous range of distinct yet nearly equivalent NL representations. While it seems difficult to construct useful semantic-distance metrics over word sequences, semantic-distance metrics in the QNR domain can be relatively smooth.[1] Ambitious examples of completion tasks could include completion of (descriptions of) code with missing functions, or of mathematical texts with missing equations or proofs.

10.5 Training QNR $\times$ QNR $\rightarrow$ QNR Functions to Respect Lattice Structure

The semantic-lattice properties discussed in Appendix A1 correspond to an algebra of information, but QNRs need not automatically respect this algebra. In particular, absent suitable training objectives, operations on QNRs may strongly violate the lattice axioms that constrain unification and generalization.[2] Learning representations and functions that approximately satisfy the lattice-defining identities (Section A1.2) can potentially act both as a regularizer and as a mechanism for training operations that support principled comparison, combination, and reasoning over QNR content.

Because the existence of (approximate) lattice operations over QNR representations implies their (approximate) correspondence to (what can be interpreted as) an information algebra, we can expect that (approximately) enforcing this constraint can improve the semantic properties of a representational system. In addition, prediction of soft-unification scores (Section A1.4.3) can provide an auxiliary training objective for content summaries (Section 8.3.4), providing a distance measure with potential applications to structuring latent spaces for similarity-based semantic retrieval (Section 9.1.2).

10.6 Processing and Inference on QNR Content

The above discussion outlined coarse-grained information flows and general training considerations using block diagrams to represent units of high-level functionality. The present section examines the potential contents of boxes

1. Aided by structural regularity (Section 8.4.) A similar approach might prove fruitful in training models that produce flat vector representations, which naturally have smooth distance metrics. This basic approach (predicting learned representations rather than raw inputs) is applied in Larsen *et al.* (2016) and related work.

2. *E.g.*, they may map approximately lattice-respecting to strongly lattice-incompatible sets of representations.

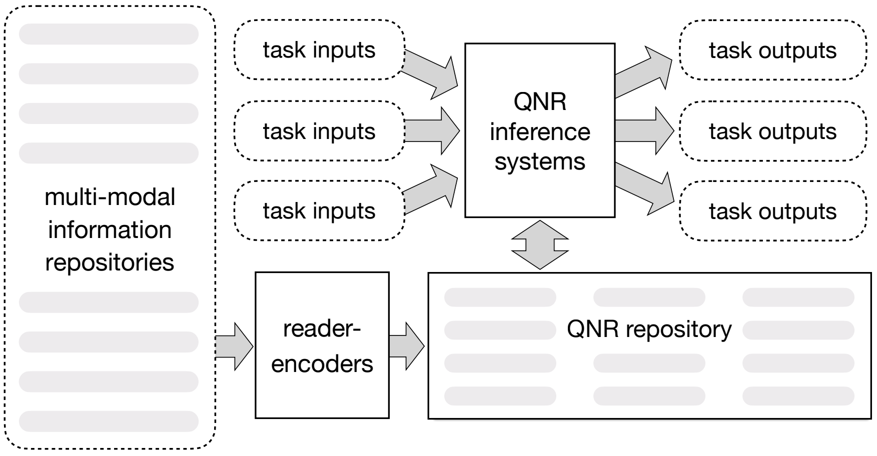

Figure 10.4: *Outline of multitask architectures that include access to external and QNR-based information repositories (e.g., the internet). More arrows could be added.*

labeled "QNR inference". The aim here is not to specify a design, but to describe features of plausible architectures for which the implementation challenges would be of familiar kinds.

10.6.1 Control, Selection, and Routing

Tasks of differing complexity will call for different QNR inference mechanisms. The null case is the identity function, single-path pass-through in a QNR-bottleneck architecture. A more interesting case would be a single-path system that performs QNR $\rightarrow$ QNR transformations (*e.g.*, using a GNN) based on a conditioning input. More powerful inference mechanisms could perform QNR $\times$ QNR $\rightarrow$ QNR operations, potentially by means of architectures that can learn (forms of) soft unification or anti-unification.

Toward the high end of a spectrum of complexity (far from entry level!), open-ended QNR-based inference will require the ability to learn task- and data-dependent strategies for storing, retrieving, and operating on QNRs in working memory and external repositories. This complex, high-end functionality could be provided by a controller that routes QNR values to operators while updating and accessing QNR values by means of key and query based

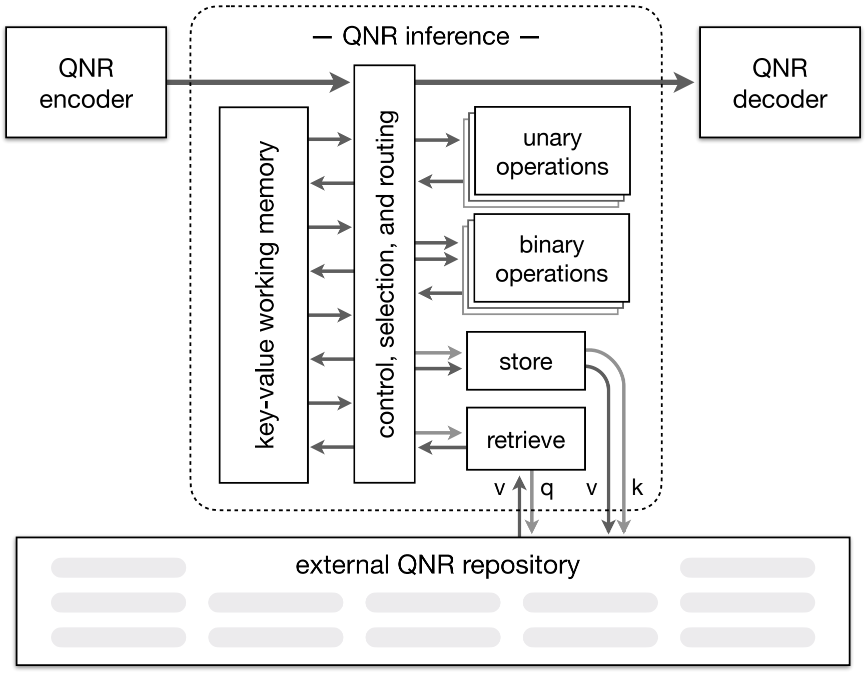

Figure 10.5: *Block diagram decomposing aspects of architectures for complex, open-ended QNR inference functionality. Both working memory and an external repository store key-value pairs, and given a query, will return one or more values associated with near-neighbor keys in a semantic embedding space. Arrows labeled q and k (shown explicitly in connection with external operations) represent query and key embeddings used in storing and retrieving QNR values (v). An elaboration of Figure 10.4 would show similar QNR inference functionality in connection, not only with "QNR inference systems", but also with "reader-encoders" (which need not be distinct components).*

storage and retrieval.[1] Keys and queries, in turn, can be products of abstractive operations on QNRs. (In discussions of retrieval, argument passing, *etc.*, "a QNR" is operationally a reference to a node in a graph that may be of indefinite extent.)

Note that "reasoning based on QNRs" can employ reasoning *about QNR processing* by means of differentiable mechanisms that operate on flat vector representations in a current task context.[2] Reinforcement learning in conjunction with memory retrieval has been effective in multi-step reasoning (Banino *et al.* 2020), as have models that perform multi-step reasoning over differentiable representations and retrieve external information to answer queries (Bauer, Wang, and Bansal 2018).

10.6.2 Encoders

QNR encoders accept task inputs (word sequences, images, *etc.*) and produce sparse-graph outputs. Natural implementation choices include Transformer-like attention architectures that initially process information on a fully connected graph (the default behavior of attention layers) but apply progressively sharpened gating functions in deeper layers. Gating can differentiably weight and asymptotically prune arcs to sparsen graphs that can then be read out as discrete structures. A range of other methods could be applied to this task.

Optional discretization at a sparse-graph readout interface breaks differentiability and cannot be directly optimized by gradient descent. This difficulty has been addressed by means that include training-time graph sampling with tools from reinforcement learning (Kazi *et al.* 2020) and other mechanisms that learn to discretize or sparsen through supervision from performance on downstream tasks (Malinowski *et al.* 2018; Zheng *et al.* 2020). Systems with potentially relevant mechanisms learn dynamic patterns of connectivity on sparse graphs (Veličković *et al.* 2020) and address problems for which the solution space consists of discrete graphs (Cappart *et al.* 2021). Because

1. In considering how this functionality might be structured, analogies to computer architectures (both neural and conventional) may be illuminating. For example, analogies with stored-program (*i.e.*, virtually all) computers suggest that memory stores can usefully contain QNRs that describe executable inference procedures. See related work in Gulcehre *et al.* (2018), Le, Tran, and Venkatesh (2020), and Malekmohamadi, Safi-Esfahani, and Karimian-kelishadrokhi (2020).

2. A psychological parallel is the use of general, fundamental "thinking skills" in reasoning about declarative memory content. Skills in this sense can be implicit in a processing mechanism (an active network rather than a repository) and are applied more directly than explicit plans.

arcs in QNRs can define paths for information flow in computation (*e.g.*, by graph neural networks), methods for training computational-graph gating functions in dynamic neural networks[1] are potentially applicable to learning QNR construction.

10.6.3 Decoders

Standard differentiable neural architectures can be applied to map QNRs to typical task-domain outputs. A natural architectural pattern would employ GNNs to process sparse graphs as inputs to downstream Transformer-like attention models. Where the intended output is fluent natural language, current practice suggests downstream processing by large pretrained language models adapted to conditional text generation;[2] potentially relevant examples include models that condition outputs on sentence-level semantic graphs.[3]

10.6.4 Working and External Memory Stores

Working memory and external repositories have similar characteristics with respect to storage and retrieval, but differences in scale force differences in implementation. In particular, where stores are large, computational considerations call for storage that is implemented as an efficient, scalable, potentially shared database that is distant (in a memory-hierarchy sense) from task-focused computations.[4] In the approach suggested here, both forms of storage would, however, retrieve values based on similarity between key and query embeddings.

10.6.5 Unary Operations

Unary operations apply to single graphs. Popular node-convolutional GNNs use differentiable message-passing schemes to update the attributes of a graph, and can combine local semantic information to produce context-informed representations. Different networks could be applied to nodes of different semantic types. The values returned by unary operations may be QNRs or embeddings (*e.g.*, keys, queries, or abstractive summaries).

1. Reviewed in Han *et al.* (2021)
2. *E.g.*, Keskar *et al.* (2019).
3. *E.g.*, Mager *et al.* (2020).
4. Fu *et al.* (2018), J. Wang *et al.* (2018), Johnson, Douze, and Jégou (2019), and Jayaram Subramanya *et al.* (2019)

Unary operations may also transform graphs into graphs of a different topology by pruning arcs (a local operation), or by adding arcs (which in general may require identifying and linking potentially remote nodes).[1] Examples of neural systems with the latter functionality were noted above.[2] A local topology-modifying operation could (conditionally) pass potential arcs (copies of local references) as components of messages.[3]

10.6.6 Graph Alignment

Graph alignment ("graph matching") is a binary operation that accepts a pair of graphs as arguments and (when successful) returns a graph that represents a (possibly partial) node-correspondence relationship between them (Section 7.2.4). Return values could range in form from a node that designates a pair of corresponding nodes in the arguments, to a representation that includes a distinguished set of arcs (potentially labeled with vector embeddings) that represent relationships among all pairs of corresponding nodes.

Several neural matching models have been demonstrated, some of which are relatively scalable.[4] Graph alignment could be a pretrained and fine-tuned function.

10.6.7 Lattice Operations

Lattice operations (unification and generalization, Appendix A1). are binary operations that include mechanisms for graph alignment and combination. Soft lattice operations differ from matching in that they return what is semantically a single graph. Like graph alignment, lattice operations could be pretrained and fine-tuned, or could serve as auxiliary training tasks in learning QNR inference. Neural modules pretrained to mimic conventional

1. A related operation would accept a graph referenced at one node and return a graph referenced at another, representing the result of a graph traversal.

2. Veličković *et al.* (2020) and Cappart *et al.* (2021)

3. This is the fundamental topology-modifying operation employed by object capability systems (Noble *et al.* 2018): A node *A* with message-passing access to nodes *B* and *C* can pass its node-*B* access (a "capability") to node *C*; node *A* may or may not retain its access to *C* afterward. This operation can be iterated to construct arcs between what are initially distant nodes in a graph. Intuitively, access-passing is semantically well motivated if the "need" for a more direct connection from *B* to *C* can be communicated through messages received by *A*. See also Veličković *et al.* (2020).

4. Y. Li *et al.* (2019), Sarlin *et al.* (2020), Y. Bai *et al.* (2020), and Fey *et al.* (2020)

algorithms for unification and generalization could potentially serve as building blocks for a range of inference algorithms that operate on soft lattices and rich semantic representations.[1]

11 Potential Application Areas

Potential applications of QNR/NL$^+$ functionality include and extend applications of natural language. They include human-oriented NLP tasks (translation, question answering, semantic search), but also inter-agent communication and the integration of formal and informal representations to support science, mathematics, automatic programming, and AutoML.

QNR/NL$^+$ frameworks are intended to support wide-ranging applications both within and beyond the scope of natural language. The present section sketches several potential application areas: first, applications to tasks narrowly centered on language—search, question answering, writing, translation, and language-informed agent behavior—and then a range of applications in science, engineering, mathematics, software, and machine learning, including the general growth and mobilization of knowledge in human society. The discussion will assume success in developing high-level QNR/NL$^+$ capabilities.

11.1 Language-Centered Tasks

Tasks that map NL inputs to NL outputs are natural applications of NL$^+$-based models. These tasks include internet search, question answering, translation, and writing assistance that ranges from editing to (semi)autonomous content creation.

11.1.1 Search and Question Answering

In search, NL$^+$ representations can provide a semantic bridge between NL queries and NL documents that employ different vocabularies. Search and question-answering (QA) models can jointly embed queries and content, enabling retrieval of NL content by semantic similarity search[2] anchored in the

1. See Veličković and Blundell (2021) and included references.
2. Reviewed inYe Zhang *et al.* (2017) and Mitra and Craswell (2018).

NL^+ domain; beyond comparing embeddings, direct NL^+ to NL^+ comparisons can further refine sets of potential search results.

Alternatively, language models conditioned on queries (and potentially on models of readers' style preferences) can translate retrieved NL^+ semantic content to fluent NL answers. QA fits well with document search, as illustrated by Google's information boxes: The response to a search query can include not only a set of documents, but information abstracted from the corpus.

In a broader application, NL^+-based models could generate extended answers that are more comprehensive, more accurate, and more directly responsive to a query than any existing NL document. With the potential for dense linking (perhaps presented as in-place expansion of text and media), query-responsive information products could enable browsing of internet-scale knowledge corpora through presentations more attractive and informative than conventional web pages.

11.1.2 Translation and Editorial Support

Translating and editing, like QA, call for interpreting meaning and producing results conditioned on corpus-based content and priors. Differences include a greater emphasis on lengthy inputs and on outputs that closely parallel those inputs, with access to specific knowledge playing a supporting rather than primary role. During training, massively multilingual translation tasks have produced language-invariant intermediate representations (interlinguas[1]); we can expect similar or better interlingua representations—and associated translations—in systems that employ $NL \rightarrow NL^+ \rightarrow NL$ architectures. Priors based on the frequency of different patterns of semantic content (not phrases) can aid disambiguation of NL source text.

The task of machine-aided editing is related to translation: Reproducing semantic content while translating from language to language has much in common with transforming a rough draft into a refined text; stretching the notion of reproducing semantic content, a system might expand notes into text while retaining semantic alignment.[2] It is again natural to exploit priors over patterns of expression to help interpret inputs and generate outputs. Access to knowledge from broad, refined corpora could greatly enrich content when expanding notes. The graph structure of hypertext makes QNRs a good fit to NL^+-supported authoring of online NL content.

1. See Y. Lu *et al.* (2018) and Arivazhagan *et al.* (2019).

2. A limiting case of this task would be semi-autonomous production of content, potentially on a large scale, guided by only the most general indications of purpose; see Section 12.2.

As a specific, high-leverage application, such tools could help contributors expand and improve Wikipedia content. Systems that compile, refine, and access a QNR translation of Wikipedia would be a natural extension of current research on the use of external information stores.[1] Human contributors could play the roles that they do today, but aided by generative models that draw on refined NL$^+$ corpora to suggest corrected and enriched content.

Most of what people want to express either repeats what has been said elsewhere (but rephrased and adapted to a context), or expresses novel content that parallels or merges elements of previous content. Mechanisms for abstraction and analogy, in conjunction with examples and priors from existing literatures, can support interactive expansion of text fragments and hints to provide what is in effect a more powerful and intelligent form of autocomplete.

Similar functionality can be applied at a higher semantic level. Responsible writers seek to avoid factual errors, which could be identified (provisionally!) by clashes between the NL$^+$ encoding of a portion of a writer's draft and similar content retrieved from an epistemically high-quality NL$^+$ corpus.[2] Writers often prefer, not only to avoid errors, but to inform their writing with knowledge that they do not yet have. Filling semantic gaps, whether these stem from omission or error removal, can be regarded as a completion task over abstract representations (Section 10.4.2). Semantically informed search and generative models could retrieve and summarize candidate documents for an author to consider, playing the role of a research assistant;[3] conditional language models, prompted with context and informed by external knowledge[4] could generate substantial blocks of text, playing the role of a coauthor.

11.1.3 (Semi)Autonomous Content Creation

Social media today is degraded by the influence of MIsinformed and unsourced content, a problem caused (in part) by the cost of finding good infor-

1. Verga *et al.* (2020), Guu *et al.* (2020), and Xu *et al.* (2020) discuss Wikipedia-oriented systems.

2. To enable retrieval of similar yet potentially clashing content, (some) embeddings should represent, not the *concrete* semantic content of expressions (in effect, answers to potential questions), but the *kinds of questions* that the content can answer, an important distinction noted above. Relevant clashes would then be indicated by failures of *partially successful* attempts at soft unification between new and retrieved content.

3. Because statements may provide support for other statements, providing such material is related to argumentation, where automated, corpus-based methods are an area of active research; for example, see Lawrence and Reed (2020) and Slonim *et al.* (2021).

4. A process illustrated by Xu *et al.* (2020).

mation and citing sources, and (in part) by fact-indifferent actors with other agendas. Well-informed replies are relatively costly and scarce, but mob-noise and bot-spew are abundant.

As a human-mediated countermeasure, responsible social media participants could designate targets for reply (perhaps with a hint to set direction) and take personal responsibility for authorship while relying on semi-autonomous mechanisms for producing (drafts of) content. As a fully autonomous countermeasure, bots created by responsible actors could scan posts, recognize problematic content, and reply without human intervention. Actors that control social media systems could use analogous mechanisms in filtering, where a "reply" might be a warning or deletion. Acceptable, fully effective countermeasures to toxic media content are difficult to imagine, yet substantial improvements at the margin may be both practical and quite valuable.

11.2 Agent Communication, Planning, and Explanation

Human agents use language to describe and communicate goals, situations, and plans for action; it is reasonable to expect that computational agents can likewise benefit from (quasi)linguistic communication.[1] If NL$^+$ representations can be strictly more expressive than NL, then NL$^+$ can be strictly more effective as a means of communication among computational agents.

Internal representations developed by neural RL agents provide another point of reference for potential agent-oriented communication. Some systems employ memories with distinct, potentially shareable units of information that can perhaps be viewed as pre-linguistic representations (*e.g.*, see Banino *et al.* (2020)). The limitations inherent in current RL-agent representations suggest the potential for gains from language-like systems in which the compositional elements express durable, shareable, strongly compositional abstractions of states, conditions, actions, effects, and strategies.

QNR/NL$^+$-based representations can combine unnamable, lexical-level abstractions with lexical-level elements that describe semantic roles, confidence, relative time, deontic considerations, and the like—in other words, semantic elements like those often expressed in NL by function words and TAM-C modifiers (Section 5.3.3, Section 5.3.4, and Section A3.3). The role of NL in human communication and cognition suggests that NL$^+$ representations can

1. For examples of related work, see Shah *et al.* (2018) and Abramson *et al.* (2021). Relatively simple linguistic representations have emerged spontaneously; see Mordatch and Abbeel (2018) and Lazaridou and Baroni (2020).

contribute to both inter- and intra-agent performance, sometimes competing with tightly coupled, task-specific neural representations.

Communication between humans and RL agents can benefit from language. Although reinforcement learning can enable unaided machines to outperform human professionals even in complex games,[1] human advice conveyed by NL can speed and extend the scope of reinforcement learning.[2] Conversational applications provide natural mechanisms for clarification and explanation—in both directions—across machine-human interfaces, potentially improving the human value and interpretability of AI actions.

Given suitable NL$^+$ descriptions and task-relevant corpora, similarity search could be applied to identify descriptions of similar situations, problems, and potentially applicable plans (including human precedents); mechanisms like those proposed for knowledge integration and refinement (Section 9.4) could be applied to generalize through analogy and fill gaps through soft unification. Widely used content would correspond to "common sense knowledge" and "standard practice".[3] Like natural language, NL$^+$ representations could support both strategic deliberation and concrete planning at multiple scales.

Agents with access to large knowledge corpora resemble humans with access to the internet: Humans use search to find solutions to problems (mathematics, travel, kitchen repairs); computational agents can do likewise. Like human populations, agents that are deployed at scale can learn and pool their knowledge at scale. Frequent problems will (by definition) seldom be newly encountered.

11.3 Science, Mathematics, and System Design

Although research activities in science, mathematics, engineering, and software development differ in character, they share abstract tasks that can be framed as similarity search, semantic alignment, analogy-building, clash detection, gap recognition, and pattern completion. Advances in these fields involve an ongoing interplay between:

1. Including games that require long-term planning (Vinyals *et al.* 2019; OpenAI *et al.* 2019).

2. Luketina *et al.* (2019) reviews progress and calls for "tight integration of natural language understanding into RL".

3. As noted above, it is reasonable to expect that the most general and frequently used kinds of knowledge would be encoded, not in declarative representations that enable multi-step inference, but in model parameters that enable direct decision and action; this distribution of functionality would parallel Kahneman's System-1/System-2 model of human cognition (Kahneman 2011).

- Tentative proposals (hypotheses in science, proof goals in mathematics, design concepts in engineering and software development),
- Domain-specific constraints and enablers (evidence in science, theorems in mathematics, requirements and available components in engineering and software development), and
- Competition between alternative proposals judged by task-specific criteria and metrics (maximizing accuracy of predictions, generality of proofs, performance of designs; minimizing relevant forms of cost and complexity).

These considerations highlight the ubiquitous roles of generative processes and selection criteria, and a range of fundamental tasks in science, mathematics, engineering, and software development can be addressed by generative models over spaces of compositional descriptions. These can be cast in terms of QNR affordances:

Given a problem, if a corpus of QNRs contains descriptions of related problems together with known solutions, then similarity search on problem-descriptions[1] can retrieve sets of potentially relevant solutions. Joint semantic alignment, generalization, and analogy-building within problem/solution sets then can suggest a space of alternatives that is likely to contain solutions—or near-solutions—to the problem at hand. In conjunction with an initial problem description, such representation spaces can provide priors and constraints on generative processes,[2] and generated candidate solutions can be tested against task-specific acceptance criteria and quality metrics. These considerations become more concrete in the context of specific task domains.

11.3.1 Engineering Design

[T]hink of the design process as involving, first, the generation of alternatives and, then, the testing of these alternatives against a whole array of requirements and constraints. There need not be merely a single generate-test cycle, but there can be a whole nested series of such cycles.

—Herbert Simon[3]

1. Along with filtering based on detailed comparisons.

2. Pattern completions may suggest structures; sampling guided by embeddings may suggest components.

3. Simon (1988). Note that Simon describes planning as a design process.

Typical engineering domains are strongly compositional, and aspects of compositionality—modularity, separation of functions, standardization of interfaces—are widely shared objectives that aid not only the design and modeling of systems, but also production, maintenance, and reuse of components across applications. Representations used in the design and modeling of engineering systems typically comprise descriptions of components (structures, circuits, motors, power sources. . .) and their interactions (forces, signals, power transmission, cooling. . .). In engineering practice, natural language (and prospectively, NL$^+$) is interwoven with formal, physical descriptions of system-level requirements, options, and actual or anticipated performance. As Herbert Simon has observed, design can be seen as a generate-and-test process—a natural application of generative models.[1] A wide range of proposed systems can be tested through conventional simulation.[2]

In engineering, even novel systems are typically composed (mostly or entirely) of hierarchies of subsystems of familiar kinds.[3] The affordances of QNR search and alignment are again applicable: Embedding and similarity search can be used to query design libraries that describe options at various levels of abstraction and precision; descriptions can be both physical and functional, and can integrate formal and informal information. Semantic alignment and unification provide affordances for filling gaps—here, unfilled functional roles in system architectures—to refine architectural sketches into concrete design proposals. The generation of novelty by soft-lattice generalization and combination operations (Appendix A1) could potentially enable fundamental innovation.

Because engineering aims to produce systems that serve human purposes, design specifications—requirements, constraints, and optimization criteria—must fit those purposes. The development of formal specifications is an informal process that can benefit from QNR affordances that include analogy, pattern completion, and clash detection, as well as applications of the commonsense knowledge needed to choose obvious defaults, reject obvious mistakes, and identify considerations that call for human attention.

1. See discussions in Kahng (2018), Liao *et al.* (2019), and Oh *et al.* (2019). Machine-aided interactive design (Deshpande and Purwar 2019) and imitation learning can also help to generate proposals; see Raina, McComb, and Cagan (2019), Raina, Cagan, and McComb (2019), and Ganin *et al.* (2021).

2. Or using ML-based simulation methods, which are of increasing scope and quality. In particular, advances in ML-based molecular simulation (reviewed in Noé *et al.* 2020) can be expected to facilitate molecular systems engineering.

3. Illustrated by work in Stump *et al.* (2019), Mo *et al.* (2019), and Chen and Fuge (2019).

11.3.2 Scientific Inquiry

Science and engineering often work closely together, yet their epistemic tasks are fundamentally different: Engineering seeks to discover multiple options for achieving purposes, while science seeks to discover uniquely correct descriptions of things that exist. Science and engineering intertwine in practice: Scientists exploit products of engineering (telescopes, microscopes, particle accelerators, laboratory procedures. . .) when they perform observations and experiments, while engineers engage in science when they ask questions that cannot be answered by consulting models.

Potential applications of QNR affordances in science include:

- Translating NL publications into uniform, searchable representations
- Applying unification to combine and extend partial descriptions
- Applying unification to identify clashes between descriptions
- Applying analogies from developed fields to identify gaps in new fields
- Applying analogies to suggest hypotheses that fill those gaps
- Matching experimental objectives to experimental methods
- Matching questions and data to statistical methods
- Assessing evidence with attention to consensus
- Assessing evidence with attention to consilience
- Enabling ongoing updates of inferential dependency structures

Applications like these need not automate scientific judgment: To provide value, they need only provide useful suggestions to human scientists. Developments along these lines would extend current directions in applying ML to scientific literatures.[1]

11.3.3 Mathematics

You have to guess a mathematical theorem before you prove it; you have to guess the idea of the proof before you carry through the details. You have to combine observations and follow analogies; you have to try and try again.

—*George Pólya*[2]

In mathematical applications, proposed QNR/NL$^+$ frameworks could wrap formal, symbolic structures in soft descriptions[3] that can be applied to help

1. *E.g.*, M. Jiang *et al.* (2020) and Raghu and Schmidt (2020)
2. Pólya (1990)
3. Szegedy (2020) suggests deriving formal expressions from NL text.

recognize analogies and express purpose, and these capabilities can operate at multiple levels of granularity. Pólya observes that discovery in mathematics involves generate-and-test cycles guided by soft considerations, and modern deep learning confirms the value of soft matching in guiding theorem proving. Better neural representations can improve ML-informed premise selection,[1] slowing the explosive growth of deep proof trees by improving the success rate of generate-and-test cycles. Graph neural networks that operate on syntactic structures can provide useful embeddings (M. Wang *et al.* 2017), and enriching formal symbolic representations with soft semantic descriptions (*e.g.*, of known use-contexts) should enable further gains. Pólya emphasizes the importance of analogy, a kind of soft, structured generalization (Section A1.1.2). The formal (hence more restrictive) lattice operation of generalization by anti-unification has been applied to analogical reasoning in symbolic mathematics (Guhe *et al.* 2010); embedding symbolic structures in soft representations could extend the scope of potential generalizations.

11.4 Software Development and AutoML

Applications of neural ML to software development are under intense exploration.[2] Language models can support interactive, text-based code completion and repair;[3] recent work has demonstrated generation of code based on docstrings.[4] GNNs could operate on structured representations (syntactic and semantic graphs) while also exploiting function names, variable names, comments, and documentation as sources of information and targets for prediction in representation encoding and decoding. QNRs can provide affordances for enriching syntactic structures with semantic annotations and the results of static program analysis,[5] and for wrapping code objects (both implemented and proposed) in descriptions of their requirements and functionality.

1. See Kucik and Korovin (2018), Bansal *et al.* (2019), and Ferreira and Freitas (2020).

2. See for example Polosukhin and Skidanov (2018), Camacho and McIlraith (2019), Wang and Christodorescu (2019), and Odena *et al.* (2020). IBM recently released a training set that includes 14 million code samples comprising about 500 million lines of code (Puri 2021).

3. W. Wang *et al.* (2020), Feng *et al.* (2020), Tarlow *et al.* (2020), and Svyatkovskiy *et al.* (2021)

4. Transformer-based models trained on GitHub Python code are good enough to be of practical value, but they are error-prone and success rates decline exponentially with increasing docstring length Chen *et al.* (2021); worse, 40% of the code has been found to contain potentially exploitable bugs (Pearce *et al.* 2021).

5. A thorough exploitation of pre-processing in the symbolic domain would provide neural networks with graph-structured inputs that encode not only syntax trees, but data structures, inferred types, data flow, and flow of control. See Allamanis, Brockschmidt, and Khademi (2018), Cummins *et al.* (2020), and Guo *et al.* (2021), and the discussion in Tarlow *et al.* (2020).

QNR representations have a different (and perhaps closer) relationship to automated machine learning (AutoML[1]), because neural embeddings and graphs seem particularly well-suited to representing the soft functionality of neural components in graph-structured architectures. Again, generate-and-test processes guided by examples, analogies, and pattern completion could inform search in design spaces,[2] while the scope of these spaces can embrace not only neural architectures, but their training methods, software and hardware infrastructures, upstream and downstream data pipelines, and more.

12 Aspects of Broader Impact

> The breadth of potential applications of QNR-based systems makes it difficult to foresee (much less summarize) their potential impacts. Leading considerations include the potential use and abuse of linguistic capabilities, of agent capabilities, and of knowledge in general. Systems based on QNR representations promise to be relatively transparent and subject to correction.

Potential roles for QNR/NL$^+$-enabled capabilities are extraordinarily broad, with commensurate scope for potential benefits and harms.[3] Channels for potential QNR/NL$^+$ impacts can be loosely divided into core semantic functionalities (applications to knowledge in a general sense), semantic functionalities at the human interface (processing and production of natural language content), and potential roles in AI agent implementation and alignment. Most of the discussion here will be cast in terms of the NL$^+$ spectrum of potential QNR functionality.

12.1 Broad Knowledge Applications

Many of the potential benefits and harms of QNR/NL$^+$-enabled developments are linked to large knowledge corpora and their applications. Several areas of potential impact are closely related to proposed core functionalities of knowledge integration and access.

1. Real *et al.* (2020) and He, Zhao, and Chu (2021)
2. You, Ying, and Leskovec (2020), Radosavovic *et al.* (2020), and Ren *et al.* (2021)
3. For a survey of a range of potential harms, see Brundage *et al.* (2018).

12.1.1 Integrating and Extending Knowledge

Translation of content from NL corpora to corresponding NL$^+$ can enable the application of QNR-domain mechanisms to search, filter, refine, integrate, and extend NL-derived content, building knowledge resources for wide-ranging applications. To the extent that improving the quality of knowledge is on the whole beneficial (or harmful), we should expect net beneficial (or harmful) impacts.

12.1.2 Mobilizing Knowledge

Translation of NL expressions (statements, paragraphs, documents. . .) to corresponding NL$^+$ representations promises to improve semantic embeddings and similarity search at scale (Section 9.1.5), helping search systems "to organize the world's information and make it universally accessible and useful" (Google 2020) through higher-quality semantic indexing and query interpretation. Generation of content through knowledge integration could go beyond search to deliver information that is latent (but not explicit) in existing corpora. It is reasonable to expect beneficial first-order impacts.

12.1.3 Filtering Information

To the extent that NL $\rightarrow$ NL$^+$ translation is effective in mapping between NL content and more tractable semantic representations, filtering of information[1] in the NL$^+$ domain can be used to filter NL sources. Potential applications span a range that includes both reducing the toxicity of social media and refining censorship in authoritarian states. In applications of language models, filtering based on disentangled representations of knowledge and outputs could mitigate leakage of private information.[2]

12.1.4 Surveillance and Intelligence Analysis

Surveillance and intelligence analysis are relatively direct applications of QNR-enabled knowledge mobilization and integration, and the balance of impacts on security, privacy, and power relationships will depend in part on how

1. *E.g.*, based on multi-source consistency, consensus, coherence, consilience, and provenance (Section 9.5.2). Current filtering methods appear to rely heavily on judgments of source quality (a domain-insensitive, non-content-based proxy for epistemic reliability), perhaps the simplest use of provenance.

2. A problem discussed in Carlini *et al.* (2021).

information is filtered, shared, and applied. Mapping raw information into structured semantic representations could facilitate almost any application, with obvious potential harms. To mitigate harms, it will be important to explore how filtering of raw information could be applied to differentially enable legitimate applications: For example, disentangled compositional representations could be more easily redacted to protect sensitive information while providing information necessary for legitimate tasks.

12.2 Producing QNR-Informed Language Outputs at Scale

We should expect to see systems that translate NL$^+$ content into NL text[1] with fluency comparable to models like GPT-3, and do so at scale. Automated, NL$^+$-informed language production, including support for human writing (Section 11.1.2), could expand quantity, improve quality, and customize the style and content of text for specific groups or individual readers. These capabilities could support a range of applications, both beneficial and harmful.

12.2.1 Expanding Language Output Quantity

Text generation enabled by language models has the potential to produce tailored content for social media economically and at scale: Human writers are typically paid ~0.20 US$/word,[2]) about 1,000,000 times the cost of querying an efficient Transformer variant.[3] It is reasonable to expect that NL$^+$-informed outputs will have broadly similar costs, orders of magnitude less than the costs of human writing, whether these costs are counted in money or time. Put differently, text output per unit cost could be scaled by a factor on the rough order of 1,000,000. Even when constrained by non-computational costs and limitations of scope, the potential impact of automated text generation is enormous.

12.2.2 Improving Language Output Quality

Applications of language-producing systems will depend in part on domain-dependent metrics of output quality: Higher quality can both expand the scope of potential applications and decrease the costs of human supervision,

1. Translation would be subject to semantic imprecision due to differences in expressive capacity.

2. Approximately—range of compensation is substantial (*e.g.,* see Tee (2021).

3. An optimized ("FastFormer") model derived from BERT can perform inference at a cost of about 18 US$/100 million queries (Kim and Hassan 2020).

while changing the nature and balance of potential impacts. Relative to opaque language models, systems informed by NL$^+$ corpora can improve abilities:

- To judge, incorporate, and update factual content
- To perform multi-step, multi-source inference
- To apply inference to refine and expand knowledge stores

Current models based on opaque, learned parameters have difficulties in all these areas; overcoming these difficulties could greatly expand the scope of potential applications.

12.2.3 Potentially Disruptive Language Products

The most obvious societal threats from NL$^+$-based language capabilities stem from their ability to produce coherent content that draws on extensive (mis)information resources—content that mimics the markers of epistemic quality without the substance. The magnitude of this threat, however, must be judged in the context of other, broadly similar technologies.

Systems based on large language models are becoming fluent and potentially persuasive while remaining factually unreliable: They can more easily be applied to produce plausible misinformation than informed content. Unfortunately, the current state of social media suggests that fluent, persuasive outputs based on false, incoherent information—whether from conspiracy fans or computational propaganda—can be disturbingly effective in degrading the epistemic environment.[1] This suggests that the marginal harms of making misinformation more coherent, better referenced, *etc.*, may be relatively small.[2] To the extent that capabilities are first deployed by responsible actors, harms could potentially be mitigated or delayed.

1. Existing language models have spurred concerns regarding abuse, including scaling of social-engineering attacks on computer security and of computational propaganda in public discourse (See Woolley and Howard (2017) and Howard (2021)). In part as a consequence of such concerns (Solaiman *et al.* (2019), and Brown *et al.* (2020), Section 6.1), OpenAI restricted access to its GPT-3 model.

2. One may hope that influential audiences that have in the past been susceptible to documents with misleading but apparently high-quality content (*e.g.*, academics and policymakers) would also respond to *prompt, well-targeted, high-quality* critiques of those documents. Responding promptly would leave less time for misleading information to spread unchecked and become entrenched as conventional wisdom.

12.2.4 Potentially Constructive Language Products

Generating low-quality content is easy for humans and machines, and arguments (whether for bad conclusions or good) can cause collateral damage when they inadvertently signal-boost false information; conversely, arguments (regardless of the merits of their conclusions) can produce what might be described as "positive argumentation externalities" when their content signal-boosts well-founded knowledge Although the potential harms of facilitating the production of (apparently) high-quality misinformation may be marginal, the potential benefits of facilitating the production of high-quality information seem large.

It would be difficult to exaggerate the potential value of even moderate success in damping pathological epistemic spirals and enabling information to gain traction based on actual merit. Authors who employ freely available tools to produce better-written, better-supported, more abundant content (drawing audiences, winning more arguments) could raise the bar for others, driving more widespread adoption of those same tools. Epistemic spirals can be positive.[1]

12.3 Agent Structure, Capabilities, and Alignment

Section 11.2 discussed NL^+ representations as potential enablers for agent performance—for example, by supporting the composition of plan elements, retrieval of past solutions, and advice-taking from humans. In considering potential impacts, opportunities for improving transparency and alignment become particularly important.

12.3.1 Agent Structure

A long-standing model of advanced AI capabilities takes for granted a central role for general, unitary agents, often imagined as entities that learn much as a human individual does. The AI-services model[2] challenges this assumption, proposing that general capabilities readily could (and likely will) emerge through the expansion and integration of task-oriented services that—crucially for potential generality—can include the service of developing new services.

In the AI-services model, broad knowledge and functionality need not be concentrated in opaque, mind-like units, but can instead emerge through

1. Effective altruists please take note.
2. Drexler (2019)

aggregation over large corpora of knowledge and tools, potentially informed both by pre-existing human-generated corpora and by massively parallel (rather than individual) experience of interaction with the world. The AI-services model fits well with the QNR/NL$^+$ model of scalable, multimodal knowledge aggregation and integration.

12.3.2 Agent Capabilities

Also in alignment with the AI-services model of general intelligence, the ability of relatively simple agents to access broad knowledge and tool sets[1] could amplify their capabilities. This prospect lends credence to long-standing threat models in which agents rapidly gain great and potentially unpredictable capabilities; the mechanisms differ, but the potential results are similar.

Classic AI-risk scenarios commonly focus on AI capabilities that might emerge from an immense, opaque, undifferentiated mass of functionality, a situation in which agents might pursue unexpected goals by unintended means. It may be safer to employ task-oriented agents (and compositions of agents) that operate within action- and knowledge-spaces that are better understood and do not grossly exceed task requirements.[2] Basing functionality on bounded, differentiated resources provides affordances for observing "what a system is thinking about" and for constraining "what an agent can know and do", potentially powerful tools for interpreting and constraining an agent's plans.[3] Accordingly, developers could seek to bound, shape, and predict behaviors by exploiting the relative semantic transparency of proposed QNR/NL$^+$ corpora to describe and structure the knowledge, capabilities, constraints, and objectives of task-oriented agents.

12.3.3 Agent Alignment

Many of the anticipated challenges of aligning agents' actions with human intentions hinge on the anticipated difficulty of learning human preferences.[4] The ability to read, interpret, integrate, and generalize from large corpora of human-generated content (philosophy, history, news, fiction, court records, discussions of AI alignment. . .) could support the development of richly

1. *E.g.*, through internet-scale search and cloud services.

2. An application of the "Principle of Least Privilege" in system design.

3. See Drexler (2019, Section 9.7).

4. Bostrom (2014) and Russell (2019)

informed models of human preferences, concerns, ethical principles, and legal systems—and models of their ambiguities, controversies, and inconsistencies.[1]

Conversational systems could be used to test and refine predictive models of human concerns by inviting human commentary on actual, proposed, and hypothetical actions. NL$^+$ systems that fulfill their promise could model these considerations more effectively than human language itself, in a way that is not fully and directly legible, yet open to inspection though the windows of query and translation.

13 Conclusions

> Current neural ML capabilities can support the development of systems based on quasilinguistic neural representations, a line of research that promises to advance a range of research goals and applications in NLP and beyond.

Natural language (NL) has unrivaled generality in expressing human knowledge and concerns, but is constrained by its reliance on limited, discrete vocabularies and simple, tree-like syntactic structures. Quasilinguistic neural representations (QNRs) can generalize NL syntactic structure to explicit graphs (Section 8.1) and can replace discrete NL vocabularies with vector embeddings that convey richer meanings than words (Section 5.3, Section 8.2). By providing affordances for generalizing and upgrading the components of NL—both its structure and vocabulary—QNR systems can enable neural systems to learn "NL$^+$" representations that are strictly more expressive than NL.

Machines with human-like intellectual competence must be fully literate, able not only to read, but to write things worth reading and retaining as contributions to aggregate knowledge. Literate machines can and should employ machine-native QNR/NL$^+$ representations (Section 8) that are both more expressive and more computationally tractable than sequential, mouth-and-ear oriented human languages.

Prospects for QNR/NL$^+$ systems make contact with a host of fields. These include linguistics (Section 5), which offers insights into the nature of expressive constructs in NL (a conceptual point of departure for NL$^+$), as well as

1. Along lines suggested by Stuart Russell (Wolchover 2015); see also discussion in Drexler (2019, Section 22).

current neural ML, in which vector/graph models and representation learning provide a concrete basis for potential QNR implementations (Section 10). Considerations that include local compositionality (Section 4.3) suggest that vector/graph constructs can provide computationally tractable representations of both complex expressions and the contexts in which they are to be interpreted (Section 8.3).

Drawing on existing NL corpora, QNR-based systems could enable the construction of internet-scale NL$^+$ corpora that can be accessed through scalable semantic search (Section 9.1), supporting a powerful ML analogue of long-term memory. In addition, QNR/NL$^+$ frameworks can support unification and generalization operations on (soft, approximate) semantic lattices (Appendix A1), providing mechanisms useful in knowledge integration and refinement (Section 9.4, Section 9.5).

Applications of prospective QNR/NL$^+$ functionality could support not only epistemically well-informed language production (Section 11.1), but the growth and mobilization of knowledge in science, engineering, mathematics, and machine learning itself (Section 11.3). The fundamental technologies needed to implement such systems are already in place, incremental paths forward are well-aligned with research objectives in ML and machine intelligence, and their potential advantages in scalability, interpretability, cost, and epistemic quality position QNR-based systems to complement or displace current foundation models (Bommasani *et al.* 2021) at the frontiers of machine learning.

Acknowledgements

I would like to thank the many individuals at the Future of Humanity Institute, DeepMind, and elsewhere for comments and questions that helped to shape this work. Special thanks go to Devang Agrawal, Teddy Collins, Owen Cotton-Barrett, Andrew Critch, David Dalrymple, Owain Evans, Aleš Flídr, Irina Higgins, Alexander Lerchner, Adam Marblestone, Michael Nielsen, Andrew Trask, Jonathan Uesato, and Dani Yogatama for helpful inputs, to Rosa Wang for many discussions and unflagging support, and to Nick Bostrom and the Future of Humanity Institute for providing the intellectual environment and support that made this work possible.

Regarding the document itself, I thank Tanya Singh for launching the LaTeX conversion project, Jimmy Rintjema for doing the work, and the Berkeley Existential Risk Initiative (BERI) for paying the bills. Finally, measures taken to counter the spread of SARS-CoV-2 deserve a measure of credit (and perhaps blame) for the scale and scope of this document.

A1 Unification and Generalization on Soft Semantic Lattices

QNR representations can support operations that combine, contrast, and generalize information. These operations—soft approximations of unification and anti-unification—can be used to implement continuous relaxations of powerful mechanisms for logical inference.

A range of formal representations of meaning, both in logic and language, have the structure of mathematical lattices. Although the present proposal for QNR systems (and aspirational NL$^+$ systems) explicitly sets aside the constraint of formality, *approximate* lattice structure emerges in NL and will (or should, or readily could) be a relatively strong property of QNRs/NL$^+$. Because lattices can provide useful properties, it is worth considering the potential roles and applications of lattice structure in QNR-based systems.

Note that the fundamental goals of NL$^+$—general superiority to NL in expressiveness and computational tractability—do not require lattice properties beyond those that NL itself provides. The ability to provide stronger lattice properties is a potential (further) strength of NL$^+$, not a requirement. In other words, lattice properties are natural and useful, yet optional.

The following sections begin by discussing the motivation for considering and strengthening lattice properties—supporting meet and join, a.k.a. unification and generalization—in light of their potential roles and utility. A brief review of approximate lattice structure in NL provides an initial motivation for applying lattice structure in NL$^+$ within the scope of a methodology that avoids commitment to formal models. Consideration of lattices in logic and in constraint logic programming further motivates the pursuit of approximations, and introduces a discussion, in part speculative, regarding inductive bias and prospective, emergent lattice-oriented QNR representations.

This topic is adjacent to many others, creating a large surface area that precludes any compact and comprehensive discussion relationships to existing work.[1] A sketch of these relationships and pointers into relevant literatures provide starting points for further reading.

1. In particular, studies of lattice semantics in NL, unification and generalization in symbolic computation, and lessons learned in the broader study of neuro-symbolic ML.

A1.1 Motivation

Typical expressions can be regarded as approximate descriptions of things, whether ambiguous (pet, rather than cat) or partial (grey cat, rather than big grey cat). Given two expressions, one may want to combine them to form either a narrower description (by combining their information) or a broader description (by combining their scope). Although many other operations are possible (averaging, perhaps, or extrapolation), narrowing and broadening are of fundamental importance, and in many semantic domains, quite useful. They can be construed as operations on a formal or approximate semantic lattice.

A1.1.1 Why Unification (Meet, Intersection, Narrowing, Specialization)?

In symbolic logic, expressions correspond to points in a lattice (defined below), and *unification* is an operation that combines two expressions to form a more specific expression by combining their compatible information.[1] In generic lattices, the corresponding operation is termed *meet*, which in many contexts can be regarded as an intersection of sets or regions. Unification combines compatible information; failures of unification identify clashes.[2] The Prolog language illustrates how unification and failures can enable reasoning and proof.

A1.1.2 Why Anti-Unification (Join, Union, Broadening, Generalization)?

Alternatively, two expressions may provide (partial) descriptions of two entities of the same kind. Here, a natural goal is to describe properties common to all things of that kind; clashes between aspects of their descriptions indicate that those aspects are not definitional.

In a lattice of expressions, this form of generalization is termed *anti-unification*,[3] which increases generality by discarding clashing or unshared information. In generic lattices, the corresponding operation is termed *join*,

1. More precisely, unification is an operation that yields the most general instance of such an expression. For an application-oriented overview, see Knight (1989).

2. When unification attempts to combine partial information *about a single entity*, it is natural for inconsistent information to imply failure.

3. More precisely, among potential more general expressions, anti-unification yields the most specific instance.

which in many instances corresponds to a union of sets or regions.[1] Anti-unification has applications in NLP and can be used to form generalizations and analogies in mathematics.[2] In neural ML, approximations of anti-unification could potentially inform priors for a distribution of unseen instances of a class.

A1.1.3 Hypotheses Regarding Roles in QNR Processing

Considerations explored in this appendix suggest a range of hypotheses regarding potential roles for approximate lattice representations and operations in QNR processing:

- That lattice representations and operations ("lattice properties") can, in fact, be usefully approximated in neural systems.
- That learning to approximate lattice properties need not impair general representational capacity.
- That approximations of meet and join operations will have broad value in semantic processing.
- That lattice properties can be approximated to varying degrees, providing a smooth bridge between formal and informal representations.
- That lattice properties can regularize representations in ways that are useful beyond enabling approximate meet and join.
- That approximations of lattice representations and operations are best discovered by end-to-end learning of neural functions.
- That explicit, approximate satisfaction of lattice identities can provide useful auxiliary training tasks.

A1.2 Formal Definitions

A mathematical lattice is a partially ordered set that can in many instances be interpreted as representing the "inclusion" of "subsumption" of one element by another. Axiomatically, a lattice has unique meet and join operations, $\wedge$ and $\vee$: the meet operation maps each pair of elements to a unique greatest lower bound (infimum), while join maps each pair of elements to a unique least upper bound (supremum).

Like the Boolean $\wedge$ and $\vee$ operators (or the set operators $\cap$ and $\cup$), meet and join are associative, commutative, idempotent, and absorptive. A *bounded*

1. Here, *larger* sets are less specific and hence provide *less* information.

2. Guhe *et al.* (2010), Martinez *et al.* (2017), and Amiridze and Kutsia (2018)

lattice will include a unique bottom element, "⊥" (in the algebra of sets, ∅; here a universally incompatible meaning, "<nil>"), and a unique top element, "⊤" (in the algebra of sets, U; here, an all-embracing generalization, "<any>").

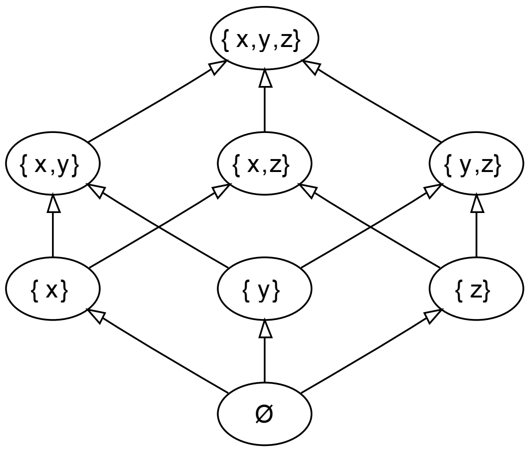

Figure A1.1: *A Boolean lattice over sets and subsets.*

In a formal infix notation, operators on a bounded lattice satisfy these identity axioms:

Idempotence:	$A \wedge A = A \vee A = A$
Commutativity:	$A \wedge B = B \wedge A$, $A \vee B = B \vee A$
Associativity:	$A \wedge (B \wedge C) = (A \wedge B) \wedge C$, $A \vee (B \vee C) = (A \vee B) \vee C$
Absorptivity:	$A \wedge (A \vee B) = A \vee (A \wedge B) = A$
Boundedness:	$A \wedge \top = A$, $A \vee \top = \top$, $A \vee \bot = A$, $A \wedge \bot = \bot$

If ∧ and ∨ are implemented as functions (rather than as structural features of a finite data structure), $A \wedge \mathrm{B} \longrightarrow C$ and $A \vee B \longrightarrow D$ will necessarily satisfy the unique-infimum and unique-supremum conditions. Commutativity can be ensured by algorithmic structure, independent from learned representations and operations; idempotence, associativity and absorptivity perhaps cannot. Boundedness is straightforward.[1]

1. Clark *et al.* (2021) provides a more extensive and formal presentation of lattice semantic structure in domains closely aligned with those considered here.

A1.3 Lattice Structure in NL Semantics

The fit between lattice orders and NL semantic structures is well known,[1] and the term "semantic lattice" has been applied not only to word and symbol-based representations, but to neural representations of images.[2] In studies of NL, both "concepts" and "properties" have been modeled in lattice frameworks, applications that speak in favor of explicit lattice structure in NL$^+$ frameworks.

A1.3.1 Models of NL Semantics: A Lattice of Concepts

Formal concept analysis[3] can recover "concept lattice" structures from text corpora, and these structures have been argued to be fundamental to information representation. Set-theoretic approaches associate formal objects with formal attributes, and construct a subsumption lattice over sets of defining attributes. Formal concept analysis has been extended to fuzzy structures in which possession of an attribute is a matter of degree.[4]

Note that lattice relationships depend on context: In the context of households, the join of "cat" and "dog" might be "pet", but in the context of taxonomy, the join would be "carnivora". In practice, expressions containing "cat" and "dog" would be considered not in isolation, but in some context; in NLP, context would typically be represented dynamically, as part of a computational state; in QNR processing, context could be included as an abstractive vector attribute (see Section 8.3.5).

A1.3.2 Models of NL Semantics: A Lattice of Properties

Properties of things may have values distributed over a continuous range, and properties associated with something may themselves specify not a precise value, but a range within the range: In NL, "light gray" does not denote a precise color, and inference from an description of a "light gray object" may only

1. "Feature structure" representations are particularly relevant to QNRs; Knight (1989) reviews feature-structure unification and generalization in NL semantics.

2. Tousch, Herbin, and Audibert (2008), Velikovich *et al.* (2018), and Wannenwetsch *et al.* (2019)

3. Cimiano, Hotho, and Staab (2005)

4. Ganter and Wille (1997, 1999), Cimiano, Hotho, and Staab (2005), Belohlavek (2011), Eppe *et al.* (2018), and Clark *et al.* (2021)

loosely constrain its reflectance. Relationships among descriptions that specify ranges of properties may correspond to an interval lattice (Section A1.5.3).[1]

A1.4 Logic, Constraint Systems, and Weak Unification

The previous section focused on lattice relationships among individual entities, but such entities can also serve as attributes in expressions or general graphs, enabling the definition of expression-level lattice operations. Lattices over expressions in which attributes themselves have non-trivial lattice orders can provide powerful, tractable representations in logic and constraint programming. Computation over such representations can be extended to include weak unification.

A1.4.1 Logical Expressions, Logic Programming

Logic programming performs reasoning based on syntactic unification of expression-trees in which symbols represent attributes of leaf nodes (variables or constants) or interior nodes (*e.g.*, functions, predicates, relations, and quantifiers). In Prolog, expressions are limited to a decidable fragment of first-order logic; more powerful unification-based systems include λ terms and can support proof in higher-order logics.[2]

Informally, first-order terms A and B unify to yield an expression C provided that all components of A (subexpressions and their attributes) unify with corresponding features or variables of B; C is the expression that results from the corresponding substitutions. Function and predicate symbols unify only with identical symbols; constants unify with variables or identical constants; variables unify with (and in the resulting expression, are replaced by) any structurally corresponding constant, variable, or subtree.[3] Aside from variables that match subtrees, tree structures must match. As required, unification of two expressions then either fails or yields the *most general expression* that *specializes* both.

Informally, expressions A and B *anti-unify*, or generalize, to yield C provided that C contains all features that A and B share, and contains variables wherever A and B differ in attributes or structure, or where either contains a variable. C is the unique, most specific expression that can unify with *any*

1. Lattices can also be constructed based on intervals with non-sharp boundaries; see Kehagias (2011) and Singh, Aswani Kumar, and Li (2016).

2. Paulson (1986) and Felty and Miller (1988)

3. Excluding subtrees that contain the same symbol when cyclic graphs are disallowed.

expression that can unify with either A or B. Thus, join/anti-unification of two expressions yields *the most specific expression* that *generalizes* both.

Relevance to QNR systems:

QNR frameworks can embed (at least) first-order logic expressions and enable their unification, provided that some attributes (representing constants, functions, *etc.*) can be compared for equality, while others (acting as variables[1]) are treated as features that match any leaf-attribute or subexpression. Accordingly, QNR frameworks augmented with appropriate algorithms can support logical representation and reasoning. This is a trivial consequence of the ability of QNRs to represent arbitrary expressions, in conjunction with freedom of interpretation and the Turing completeness of suitable neural models. Logical expressions and logic programming are, however, instances of richer systems—also within the potential scope of QNRs—that represent constraint systems and support constraint logic programming.

A1.4.2 Constraints and Constraint Logic Programming

In constraint logic programming,[2] representations are extended to include constraints more general than equality of components and binding to unconstrained variables, and unification is extended to (or replaced by) constraint satisfaction. The application of constraints narrows the variable domains, and unification fails when domains become empty. The attributes of constraint expressions have a lattice structure, as do the expressions that contain them, and constraint expressions can be narrowed and generalized though unification and anti-unification (Yernaux and Vanhoof 2019).

The potential complexity of constraints spawns a vast menagerie of constraint systems, algorithms, and constraint logic programming algorithms. Provided that expressions can include both single-element domains and variables able to bind subexpressions, constraint logic programming can subsume conventional logic programming.

Relevance to QNR systems:

As with logic and logic programming, the generality of QNR representations and neural computation implies the ability to represent constraint

1. Convergent arcs in DAG expressions model the behavior of named variables that occur in multiple locations. It should be noted that unification can produce and operate on cyclic graphs unless this is specifically excluded; see Smolka (1992).

2. Jaffar and Maher (1994)

systems and constraint-based computation, including constraint logic programming. The generalization from logic to constraints is important to semantic representational capacity: Expressions can often be interpreted as denoting regions in a semantic space,[1] and combinations of expressions can combine constraints. Constraint logic programming provides an exact formal model of computation based on this semantic foundation.

A1.4.3 Weak and Soft Lattice Operations

"Never express yourself more clearly than you are able to think"
—Niels Bohr

The literature describes a range of models of NL semantics and reasoning based on a range of approximate unification operations. These typically replace equality of constants (functions, *etc.*) with similarity: In "soft unification" (as the term is typically used in the literature) the operation succeeds if similarity (for example, cosine similarity between vectors, Arabshahi *et al.* (2021)) is above some threshold, and success may yield either conventional binding of variables to values (Campero *et al.* 2018), or merged representations of values (Cingillioglu and Russo 2020). "Weak unification" may produce a "unification score" that indicates the quality of unification; these scores can be carried forward and combined to score the quality of multi-step inference operations.[2]

As used in the present context, the term "soft unification" subsumes both "weak" and "soft" unification as used in the literature, and entails combining representations of values in a way that approximates unification operations in constraint logic programming. Thus, "softness" allows operations that violate strict lattice identities.[3] The intended class of QNR (soft-)unification

1. For example, ranges of compatible meanings with respect to various properties, implying what are in effect interval constraints on those properties, a familiar (if perhaps too rigid) constraint structure (see Benhamou 1995).

2. *E.g.*, in Sessa (2002), Medina, Ojeda-Aciego, and Vojtáš (2004), Weber *et al.* (2019), and Minervini *et al.* (2020).

3. In addition, lattice operations may be mixed: In combining information, differences of some kinds should lead to rejection or narrowing, while differences of other kinds should lead to generalization. For example if we are combining pieces of evidence about a cat (perhaps from two photographs), some properties should be unified (pictures of spots that differ only in visibility and angle of view should narrow possible models of coloration, while a difference of orange *vs.* black should lead to failure and reject the hypothesis "same cat"). By contrast, if one photo shows a sleeping cat and the other an alert cat, the combined description should represent a cat that is not always asleep. Differences between kinds of differences should be learned from and conditioned on tasks.

operations follows constraint logic programming in generalizing from the binding of constants to named (in effect, shared) unconstrained variables to include the narrowing of shared (in effect named), potentially constrained attributes. As in logic programming, QNR unification will permit the unification of subexpressions with variable-like attributes, but differs in that constrained attributes (unlike variables) may impose semantically non-trivial constraints on permissibility and on the content of resulting subexpressions (Section A1.6.4).

A1.5 Exact Lattice Operations on Regions

Although embeddings in vector/graph representations denote points in semantic spaces, their semantic *interpretations* will typically correspond to regions in lower-dimensional spaces. Comparisons to symbolic and more general constraint representations can provide insights into potential QNR representations and reasons for expecting their lattice properties to be inexact.

A1.5.1 Conventional Symbolic Expressions

In conventional expression graphs, attributes comprise symbols that represent points together with symbols that represent unbounded regions in the space of expressions. Thus, individual attribute subspaces are simple, have no spatial structure, and accordingly exhibit trivial behavior under unification and anti-unification.

A1.5.2 General Region Representations and Operations

Leaf attributes can represent regions in $\mathbb{R}^n$, but options for their unification and generalization may be representation-dependent. The conceptually straightforward definition is both trivial and problematic: Treating regions as sets of points ($A \vee B = A \cup B$) in effect discards spatial structure, and with it the potential for non-trivial generalization. Further, if region representations have limited descriptive capacity, then the result of generalizing a pair of attributes by set union cannot in general be represented as an attribute.[1]

Alternatively, the generalization of two volumes might be defined as their convex hull. Generalization of convex regions yields convex regions, and inclusion of points outside the initial volumes reflects spatial structure and a

1. Consider the union of disjoint volumes, each already at the limit of representable complexity. Intersections can (but do not necessarily) suffer from a similar difficulty.

plausible notion of semantic generalization. Unfortunately, this definition can also fall afoul of limited descriptive capacity, because the convex hull of two regions can be more complex than either.[1]

A1.5.3 Interval (Box) Representations and Operations

There are region-representations for which lattice operations are exact, for example, one-dimensional intervals in $\mathbb{R}^2$ and their generalization to axis-aligned boxes in $\mathbb{R}^n$.[3] Unification of a pair of box-regions yields their intersection; anti-unification yields the smallest box that contains both. Axis-aligned box regions can be represented by vectors of twice the spatial dimensionality (for example, by pairing interval centers with interval widths), and lattice operations yield representations of the same form. Interval-valued attributes have been applied in constraint logic programming.[4] Generalization through anti-unification of intervals has a natural semantic interpretation: Points in a gap between intervals represent plausible members of the class from which the intervals themselves are drawn.

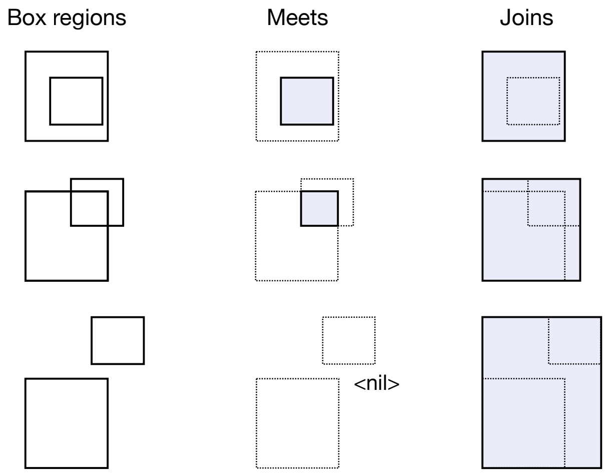

Figure A1.2: *Exact meets and joins of interval (box) regions.*

1. To illustrate, the convex hull of two spheres need not be a sphere, and the convex hull of two polytopes may have more facets than either. Intersections can also become more complex (see Jaulin 2006).

2. Discussed for example in Clark *et al.* (2021).

3. Affine (and other) transformations of a space and its regions can of course maintain these properties.

4. Benhamou and Older (1997) and Older and Vellino (1990)

A1.6 Approximate Lattice Operations on Regions

If box lattices are exact, why consider approximate operations on more general regions? The basic intuition is that representations entail trade-offs, and that greater flexibility of form is worth some degree of relaxation in the precision of unification and generalization. Boxes have sharp boundaries, flat facets, and corners; natural semantic representations may not. Intervals in a space of properties may correspond to natural semantic concepts, yet orthogonality and global alignment of axes may not.

In the present context, it is important to distinguish two kinds of approximation: As discussed in Section A3.4, effective QNR frameworks must be able to express, not only precise meanings, but ranges or constraints on meaning—a kind of approximation in the semantic domain that differs from approximation of lattice properties. Ranges of meanings can be represented as regions in semantic spaces, while region-representations that *approximate* precise meanings can *precisely* satisfy the lattice axioms.

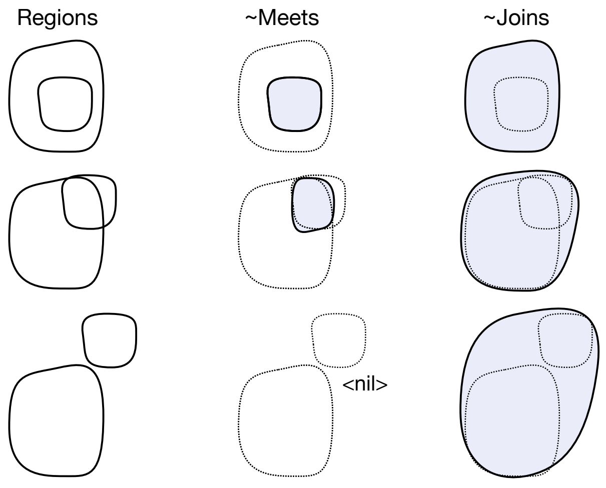

Figure A1.3: *Approximate meets and joins of regions from a less-constrained family of region shapes.*

A1.6.1 Continuous-Valued Functions

In addition, general considerations motivate representing the membership of points in semantic classes with continuous-valued functions, rather than functions having values restricted to $\{1, 0\}$. Such representations invite further approximations of lattice properties.

A1.6.2 Region Shapes and Alignment

It is natural to exploit the flexibility of neural network functions to represent generalized region shapes, and to use the ability of fully connected layers to free representations from preferred axis alignments and thereby allow exploitation of the abundance of nearly orthogonal directions in high dimensional spaces. Learned attribute representations need not describe box-like regions or share meaningful alignment in different regions of a semantic space.

A1.6.3 Satisfaction of Lattice Axioms as an Auxiliary Training Task

Given the value of lattice structure in representations, it is natural to speculate that promoting lattice structure through inductive bias may aid performance in a range of semantic tasks. Auxiliary training tasks in which losses explicitly measure violations of lattice properties may therefore be useful components of multitask learning.[1]

A1.6.4 Unifying Attributes with Expressions

As noted above, conventional symbolic expressions allow unification of unconstrained variables with subexpressions, while in the QNR context, it is natural to seek to unify subexpressions with constrained variables—attributes that represent semantic regions. The outlines of desirable behavior are clear, at least in some motivating cases:

Consider a pair of graphs with subexpressions A and B in corresponding locations. Let A be a vector representation (*e.g.*, describing a generic grey-striped animal with sharp claws), while a corresponding subexpression B is a vector/graph representation (*e.g.* that contains components that describe a cat's temperament, ancestry, and appearance). Unification should yield a vector/graph representation in which the properties described by vector A constrain related properties described anywhere in expression B (*e.g.*, the cat's appearance, paws, and some aspects of its ancestry) If some component of B specifies a black cat, unification fails. In this instance, generalization should yield a vector representation that does not clash with properties described in either A or B, while discarding properties that are not shared.

1. Note that adherence to lattice properties in representational vector spaces is important only in those regions/manifolds that are actually used for representation.

A1.6.5 Learning Lattice-Oriented Algorithms

Classic algorithms for unification and generalization implement particular patterns of information flow, intermediate representations, and iterative, conditional computation. Work on supervised neural algorithmic learning illustrates one potential approach to adapting such algorithms to neural computation, an approach that supervises the learning of algorithmic structure while allowing end-to-end learning of rich representations and corresponding decision criteria.[1]

A1.7 Summary and Conclusions

Lattice structure is found in formal systems and (to some extent) in NL, and it seems both natural and desirable in QNR/NL$^+$ representations. Although lattice structure is not a criterion for upgrading NL to NL$^+$ representations, substantial adherence to lattice structure could potentially improve expressive capacity in a systemic sense and need not imply the rigidity of fully formal representations.

Because linguistic representations and reasoning are themselves approximate, there seems little reason to sacrifice representational flexibility in order to enforce exact and universal satisfaction of the lattice axioms. A QNR framework that embraces both precise and approximate lattice relationships can enable both formal and informal applications of those relationships to formal and informal reasoning. The literatures on conventional and constraint logic programming illustrate the power and computational tractability of algorithms based on systems of this kind.

The potential benefits of approximate lattice structure may emerge spontaneously, but can also be pursued by inductive bias, including training that employs satisfaction of lattice axioms as an auxiliary task in multitask learning.

1. See Veličković and Blundell (2021) and included references.

A2 Tense, Aspect, Modality, Case, and Function Words

Tables of examples illustrate expressive constructs of natural languages that do not reduce to nouns, verbs, and adjectives.

Natural languages express a range of meanings through closed-class ("function") words, and express distinctions of tense, aspect, modality, and case though both function words and morphological features. Section 5.3 discusses the roles and importance of these constructs; this appendix provides several brief tables of examples.

Table A2.1: Classes and Examples of Function words. The examples below cover about one third of the function-word vocabulary of the English language. In strongly inflected languages, the roles of some of these function words are performed by morphological distinctions.

Determiners:	the, a, this, my, more, either
Prepositions:	at, in, on, of, without, between
Qualifiers:	somewhat, maybe, enough, almost
Modal verbs:	might, could, would, should
Auxiliary verbs:	be, do, got, have
Particles:	up, down, no, not, as
Pronouns:	she, he, they, it, one, anyone
Question words:	who, what, where, why, how
Conjunctions:	
— *coordinating:*	for, so, and, nor, but, or, yet
— *subordinating:*	if, then, thus, because, however
— *temporal:*	before, after, next, until, when, finally
— *correlative:*	both/and, either/or, not/but

Table A2.2: Examples of Tense/Aspect Distinctions. Languages can use inflection or function words to express distinctions that describe (e.g.) relative time, duration, or causation. Languages differ in the distinctions that they can compactly express, while properties like "remoteness", "completion", and "causation" invite continuous representations.)

Perfect tenses	— completed in past, present, or future ("had/has/will have" finished)
Continuous tenses	— ongoing in past, present, or future ("was/am/will be" working)
Past perfect continuous	— previously ongoing in the past ("had been working")
Future perfect continuous	— previously ongoing in the future ("will have been working")
Remote perfect	— completed in the remote past (Bantu languages[1])
Resultative perfect	— completed past action causing present state (Bantu languages)

Table A2.3: Examples of Modality Distinctions. Languages can express modalities by inflection or function words. The existence of graded degrees and overlaps within and between modalities suggests the potential value of continuous vector-space representations.

Interrogative	— Question
Imperative	— Command
Indicative	— Unqualified statement of fact
Inferential	— Qualified (inferred) fact
Subjunctive	— Tentative or potential fact
Potential	— Possible condition
Conditional	— Possible but dependent on another condition
Hypothetical	— Possible but counterfactual condition
Optative	— Desired condition
Deontic	— Ideal or proper condition

1. Bantu languages include unusually complex and expressive systems of tense and aspect (Nurse and Philippson 2006; Botne and Kershner 2008).

Table A2.4: Examples of Case Distinctions. Case distinctions can express the roles of words in a sentence (in a familiar grammatical sense) or the roles of what they denote in a situation. The number of inflectional case distinctions varies widely among languages; English has three, Tsez has dozens, many of which are locative. As with modalities, blurred boundaries and overlaps between cases suggest the potential value of continuous vector-space representations.

Nominative	— subject of a verb
Accusative	— object of a verb
Dative	— indirect object of a verb
Genitive	— relationship of possession
Comitative	— relationship of accompaniment
Lative	— movement to something
Ablative	— movement away from something
Orientative	— orientation toward something
Locative	— location, orientation, direction
Translative	— becoming something
Instrumental	— means used for an action
Causal	— cause or reason for something
Benefactive	— beneficiary of something
Terminative	— limit or goal of an action

A3 From NL Constructs to NL$^+$

Condensing, regularizing, and extending the scope of semantic representations can improve expressive capacity and compositionality, and can support theoretically grounded methods for comparing and combining semantic information.

Section 6 discussed QNR architectures as a framework, considering potential components, syntactic structures, and their semantic roles. The present section extends this discussion to explore in more detail how anticipated QNR frameworks could subsume and extend the expressive capabilities of natural languages to fulfill the criteria for NL$^+$. Key considerations include facilitating representation learning, upgrading expressiveness, improving regularity, and enabling more tractable reading, interpretation, and integration of content at scale.

Potential NL/NL$^+$ relationships discussed here are not proposals for handcrafted representations, nor are they strong or confident predictions of the results of representation learning. The aim is instead to explore the scope and strengths of QNR expressive capacity, with potential implications for design choices involving model architectures, inductive bias, and training tasks. Where neural representation learning provides NL$^+$ functionality by different means, we should expect those means to be superior.

A3.1 Upgrading Syntactic Structure

To be fit for purpose, NL$^+$ syntactic structures must subsume and extend their NL counterparts while improving computational tractability:

- To subsume NL syntactic structures, NL$^+$ frameworks can embed NL syntax; as already discussed, this is straightforward.
- To extend NL syntactic structures, NL$^+$ frameworks can support additional syntactic structure; as already discussed, this can be useful.
- To improve tractability in learning and inference, NL$^+$ frameworks can improve semantic compositionality, locality, and regularity. This potential NL/NL$^+$ differential will call for closer examination.

A3.1.1 Making Syntactic Structure Explicit

Explicit representations avoid the computational overhead of inferring structure from strings, as well as costs (and potential failures) of non-local disambiguation, *e.g.*, by using DAGs to represent coreference. Improved locality and tractability follow.

A3.1.2 Extending the Expressive Scope of Syntactic Structure

Explicit graphs can enable the use of structures more diverse and complex than those enabled by natural language and accessible to human cognition.[1] Writers grappling with complex domains may employ supplementary representations (diagrams, formal notation), or may simply abandon attempts to explain systems and relationships that involve deeply nested structures, heterogeneous patterns of epistemic qualification, and complex patterns of coreference—all of which must in NL be encoded and processed as sequences. More general representations can directly describe relationships that are more complex yet readily accessible to machines.

A3.1.3 Extending the Concept of "Syntactic Structure"

"Natural language syntax" as understood by linguists is often in practice supplemented with visually parsable structures such as nested text (outlines, structured documents) and tables that represent grids of relationships. Diagrams may embed text-labels as attributes of graphs that represent networks of typed relationships (taxonomy, control, causation, enablement, *etc.*); program code is adjacent to NL, yet goes beyond NL concepts of syntax. Implementations that support explicit bidirectional links can model the structure of literatures that support not only "cites" but also "cited by" relationships.

A3.1.4 Collapsing Syntactic Structure

In neural-network computation, vector operations are basic, while graph operations may impose additional computational overheads. This motivates the use of embeddings in preference to graph expressions,[2] which in turn

1. What is in some sense accurate linguistic encoding does not ensure successful communication: For example, as a consequence of working-memory constraints, humans suffer from "severe limitations" in sentence comprehension (Lewis, Vasishth, and Van Dyke 2006).

2. When feasible, of course. Another reason is the natural (and often semantically appropriate) commutativity implicit in vector representations viewed as sums of components.

highlights the value of highly expressive lexical-level units. In addition, encoding different meanings as differences in syntactic structure can impede alignment and comparison of expressions; When individual embeddings can express a range of meanings that NL would represent with different syntactic structures, those syntactic differences disappear.

A3.2 Upgrading Lexical-Level Expressive Capacity

Where possible, we would like to replace the syntactic compositionality of words with the simpler, arithmetic compositionality of vectors. Because the best syntax is no syntax, it is worth considering what kinds of semantic content can be represented in vector spaces without relying on graph structure.

The following discussion notes several kinds of semantic structure that exist in NL, can be represented by embeddings, and have emerged (spontaneously and recognizably) in neural models.

A3.2.1 Subsuming and Extending Content-Word Denotations

As careful writers know, there often is no single word that accurately conveys an intended meaning, while to unpack an intended meaning into multiple words may be too costly in word-count or complexity. Lexical-level embeddings can shift the ambiguity/verbosity trade-off toward expressions that are both less ambiguous *and* more concise.[1]

Embeddings can place content-word meanings in spaces with useful semantic structure.

Word embeddings demonstrate learnable structure in lexical-level semantic spaces. The geometry of learned word embeddings can represent not only similarity, but analogy,[2] and representations of semantic differentials across vocabularies[3] can be identified in language models.

Unfortunately, the role of NL word embeddings—which must represent polysemous, context-dependent words—precludes clean representation of word-independent semantic structure. Vector spaces that represent *meanings* rather than *words* can provide semantic structure that is more reliable

1. Concise expression is of course less necessary in the context of indefatigable machine intelligence.

2. As discussed in Chen, Peterson, and Griffiths (2017).

3. Daniel Kahneman's doctoral dissertation examines semantic differentials (Kahneman 1961). Semantic structure embeddable in vector spaces has been extensively studied by both linguists and ML researchers (*e.g.*, see Messick (1957), Sagara *et al.* (1961), Hashimoto, Alvarez-Melis, and Jaakkola (2016), and Schramowski *et al.* (2019)).

and useful, hence patterns observed in vector representations of natural language provide only limited insight into the potential generality and utility of semantically structured vector spaces.

Embeddings can disambiguate, interpolate, and extend vocabularies of content words.

Because points in an embedding space could be used to directly designate every word in every NL vocabulary—with vast capacity to spare—embeddings can be strictly more expressive than NL words. Again taking NL as a baseline, embeddings offer the advantage of avoiding polysemy, maladaptive word ambiguity,[1] and a large (even comprehensive?) range of recognizably missing word meanings (the frustrating-thesaurus problem). In suitable semantic spaces, points within and beyond the rough equivalents of NL word clusters can, at a minimum, interpolate and extend the rough equivalents of NL vocabularies.

Embeddings can extend vocabularies by folding modifier-expressions into noun and verb spaces.

The ability to collapse a range of multi-word expressions into embeddings is equivalent to extending vocabularies: In a straightforward example, adjectives and adverbs can modify the meanings of nouns and verbs to produce different lexical-level meanings. These modifiers often represent cross-cutting properties[2] that can describe things and actions across multiple (but not all) domains. Although words with modifiers can be viewed as extensions of NL vocabularies, expressions of limited size necessarily leave gaps in semantic space; continuous vector embeddings, by contrast, can fill regions of semantic space densely.

As a consequence of the above considerations, a function of the form NL-encode: (*lexical-NL-expression*) $\rightarrow$ (*lexical-embedding*) can exist, but not its inverse. NL-encode is neither injective nor surjective: Multiple NL expressions may be equivalent, and typical NL$^+$ expressions will have no exact NL translation.

Directions in embedding spaces can have interpretable meanings.

Linguists find that NL words can with substantial descriptive accuracy be positioned in spaces in which axes have conceptual interpretations[3] or

1. And conversely, maladaptive precision in the form of (for example) forced number and gender distinctions.

2. *E.g.*, color, mass, temperature, frequency, speed, color, loudness, age, beauty, and danger.

3. Gärdenfors (2000) and Lieto, Chella, and Frixione (2017)

reflect semantic differentials. In NL, modifiers commonly correspond to displacements with components along these same axes.

It is natural to interpret modifier-like vector components differently in different semantic domains.[1] As noted previously, the interpretation of directions in a (sub)space that corresponds to *differentials applicable to entities* would naturally depend on location—on which regions of a (sub)space correspond to which *kinds of entities*.[2] In a semantic region that describes *persons*, for example, directions in a subspace might express differences in health, temperament, age, and income, while in a semantic region that describes *motors*, directions in that same subspace might express differences in power, torque, size, and efficiency.

Lexical level and syntactic compositionality are complementary.
As suggested above, vector addition of modifiers and other differentials can express compositional semantics *within* embeddings. Exploiting this capacity does not, of course, preclude syntactic compositionality: QNRs can support compositional representations through both vector-space and syntactic structure. Without predicting or engineering specific outcomes, we can expect that neural representation learning will exploit both mechanisms.

A3.2.2 Image Embeddings Illustrate Vector Compositionality in Semantic Spaces

Image embeddings can provide particularly clear, interpretable examples of the expressive power—and potential compositionality—of continuous vector representations. (Section 9.2 discusses image and object embeddings, not as examples of representational power, but as actual lexical units.)

Humans are skilled in perceiving systematic similarities and differences among faces. Diverse architectures (*e.g.*, generative adversarial networks, variational autoencoders, and flow-based generative models[3]) can produce face embeddings that spontaneously form structured semantic spaces: These models can represent faces in high-dimensional embedding spaces that represent

1. Adjective meanings have been modeled as functions of nouns (Baroni and Zamparelli (2010); see also Blacoe and Lapata (2012)).

2. Setting aside, for the moment, useful relationships between these differentials and differences of kind. In practice, descriptions of both kinds and properties can naturally be folded into a single embedding, with no need to explicitly or cleanly factor representations into kind- and property-spaces.

3. Klys, Snell, and Zemel (2018), Kingma and Dhariwal (2018), R. Liu *et al.* (2019), and Shen *et al.* (2020)

kinds of variations that (qualitatively, yet clearly) are both recognizable and systematic.[1] Many papers present rows or arrays of images that correspond to offsets along directions in embedding spaces, and these naturally emphasize variations that can be named in captions (gender, age, affect...), but a closer examination of these images also reveals systematic variations that are less readily described by words.

Words or phrases of practical length cannot describe ordinary faces such that each would be recognizable among millions. A single embedding can.[2] Similar power can be brought to bear in a wider range of lexical-level representations.[3]

A3.3 Subsuming and Extending Function-Word/TAM-C Semantics

As noted in Section 5.3.4 TAM-C meanings can be encoded in either word morphology or function (closed-class) words. Some TAM-C modifiers are syntactically associated with lexical-level units; others are associated with higher-level constructs.

TAM-C modifiers that represent case (*e.g.*, nominative, accusative, instrumental, benefactive; see Table A2.4) can directly describe the roles of things denoted by words (*e.g.*, acting *vs.* acted upon *vs.* used), but case also can indirectly modify the meaning of a word—a rock regarded as a geological object differs from a rock regarded as a tool.[4]

Other TAM-C modifiers express semantic features such as epistemic confidence, sentiment, and use/mention distinctions. In NL, statement-level meanings that are not captured by available TAM-C modifiers may be emergent within an expression or implied by context; by compactly and directly expressing meanings of this kind, expression-level embeddings can provide affordances for improving semantic locality, compositionality, and clarity.[5]

1. A good recent example is Shen *et al.* (2020), which finds that diverse faces can be well-represented in 100-dimensional spaces (Härkönen *et al.* 2020).

2. In principle, to distinguish among millions of faces requires distinguishing on the order of 10 gradations on each of 6 dimensions, but typical embeddings are far richer in both distinctions and dimensionality.

3. Within the domain of concrete, interpretable images, ~100 dimensional embeddings can represent not only faces, but also diverse object classes and their attributes, thereby representing (in effect) interpretable "noun-and-adjective" combinations, few of which can be compactly and accurately described in NL; *e.g.*, see Härkönen *et al.* (2020).

4. Is the rock hard and sharp, or a piece of fine-grained ultramafic basalt?

5. On the internet, emoji have emerged to compactly express expression-level sentiment (*e.g.*, 😀 and 😟), and these can express meanings distributed over more than one dimension (consider 😮, 🙄, and 😉).

Some function words connect phrases: These include words and combinations that express a range of conjunctive relationships (*and, or, and/or, therefore, then, still, however, because, despite, nonetheless...*) and capability/intention related relationships (*can, could, should, would-if, could-but, would-if-could, could-but-shouldn't...*). Consideration of the roles of these words and constructs will show that their meanings are distributed over semantic spaces, and the above remarks regarding the use of embeddings to interpolate and extend NL meanings apply.[1]

In NL, tense/aspect modifiers express distinctions in the relative time and duration of events, and because these modifiers reference a continuous variable—time—they can naturally be expressed by continuous representations. Likewise, epistemic case markers in NL (indicative, inferential, potential) implicitly reference continuous variables involving probability, evidence, and causality.

Note that much of function-word/TAM-C space represents neither kinds nor properties of things, and is sparsely populated by NL expressions. The use of embeddings to interpolate and extend meanings in these abstract semantic roles could greatly improve the expressive capacity of QNR/NL$^+$ frameworks relative to natural languages.

A3.4 Expressing Quantity, Frequency, Probability, and Ambiguity

Discrete NL constructs express a range of meanings that are more naturally expressed in continuous spaces: These include number, quantity, frequency, probability, strength of evidence, and ambiguity of various kinds.

NL can express specific cardinal, ordinal, and real numbers, and absolute concepts such as *none* or *all*, but many other useful expressions are either crude (grammatical singular *vs.* plural forms) or ambiguous (*e.g., several, few, many, some, most,* and *almost all,* or *rarely, frequently,* and *almost always).* Note that *intentional* ambiguity is useful: *Few* does not denote a particular number, but a range of "small" numbers, either absolute (about 2 to 5, Munroe (2012)) or relative to expectations regarding some set of entities. This range of meanings (likewise for *unlikely, likely, possibly, almost certainly, etc.*) invites continuous representations in QNR frameworks.[2]

1. *E.g.*, embeddings that generalize NL conjunctive/causal expressions could presumably express meanings like "object *X*, (probably) *together with* and (possibly) *because* of *Y*", and do so with graded degrees of probability or epistemic confidence.

2. It is natural to want semantic spaces that express joint probability distributions as well as relationships in Pearl's do-calculus; the blurry distinction between these and the NL-like semantic spaces outlined above points to the soft boundaries of NL-centric conceptions of NL$^+$.

Similar remarks apply to qualitative and probabilistic hedges (*mostly, partially, somewhat, to some extent)* qualifiers and often agent-centered epistemic qualifiers (*presumably, if I recall correctly, it seems to me, as far as I know, in my opinion, etc.*).[1] One would also like to be able to compactly express qualifiers like *illustrative but counterfactual simplification*, *approximate description of a typical case*, and *unqualified statement but with implied exceptions:* Today, the absence of universal idioms for expressing these meanings gives rise to gigabytes of fruitless, argumentative noise in internet discussions.

The discussion above implicitly frames the expression of ambiguity (*etc.*) as a task for lexical units in phrases, following the example of NL. There are advantages, however, to folding ambiguity (*etc.*) into embeddings that represent, not *points*, but *regions* in a semantic space. A shift from point- to region-oriented semantics allows systems of representations that can approximate mathematical lattices (Appendix A1) and lattice-based inference mechanisms like Prolog and constraint logic programming (Section A1.4). These mechanisms, in turn, provide semi-formal approaches to matching, unification, and generalization of representations, with applications outlined below and explored further in Section 8.4.3.

A3.5 Facilitating Semantic Interpretation and Comparison

Relative to NL, NL$^+$ frameworks offer potential advantages that include

1. Greater expressive capacity
2. More tractable interpretation
3. More tractable comparison.

The preceding sections have discussed advantages of type (1) that stem largely from improvements at the lexical level. The present section will consider how condensation, localization, regularization of expression-level representations can provide advantages of types (2) and (3). A central theme is the use of tractable, uniform, embedding-based representations to provide expressive capacity of kinds that, in NL, are embodied in less tractable—and often irregular—syntactic constructs.

1. Expressions that are frequently reduced to abbreviations are likely to represent broadly useful, lexical-level meanings: IIRC, ISTM, AFAIK, IMO, *etc.*

A3.5.1 Exploiting Condensed Expressions

As noted above, embeddings can condense many noun-adjective, verb-adverb, and function word constructs, facilitating interpretation by making their semantic content available in forms not entangled with syntax. Further, embeddings can be compared through distance computations,[1] potentially after projection or transformation into task- and context-relevant semantic spaces. These operations are not directly available in NL representations.

A3.5.2 Exploiting Content Summaries

Content summaries (Section 8.3.4) can cache and amortize the work of interpreting expressions. The value of summarization increases as expressions become larger: An agent can read a book (here considered an "expression") to access the whole of its information, but will typically prefer a book accompanied by a summary of its topic, scope, depth, quality, and so on. Semantically optional summaries (perhaps of several kinds) can facilitate both associative memory across large corpora (Section 9.1.2) and shallow reading ("skimming") of retrieved content. Shallow reading, in turn, can enable quick rejection of low-relevance content together with fast, approximate comparison and reasoning that can guide further exploration. Where uses of information differ across a range of tasks, useful summaries of an expression may likewise differ.

A3.5.3 Exploiting Context Summaries

Although context summaries (Section 8.3.5), like content summaries, are in principle semantically redundant, they are substantially different in practice: Expressions are bounded, but an interpretive context may be of any size, for example, on the scale of a book or a body of domain knowledge. Thus, absent summarization, contextual information—and hence the meaning of an expression—may be far from local; with context summarization, meaning becomes more local and hence more strongly compositional.[2]

1. Or intersection- and union-like operations in region-oriented semantics, see Appendix A1.

2. As noted elsewhere, current language models typically encode (costly to learn, difficult to share) summaries of global context—as well as knowledge of narrower contexts and even specific facts—while their inference-time activations include (costly to infer) summaries of textually local context. Learning and sharing task-oriented summaries of both broad and narrow contexts could provide complementary and more efficient functionality. Embeddings can provide the most compact summaries, but more general QNRs could provide richer yet still abstractive information.

Some use-patterns would place *specific* semantic content in narrow context representations, and *schematic* semantic content in expressions that can contribute to descriptions in a wide range of contexts. In interpreting an expression, the effective, interpreted meanings of its embeddings would be strongly dependent on its current context. A programming language analogy would be the evaluation of expressions conditioned on binding environments, but in the QNR case, employing embeddings in place of conventional values and variables,[1] and employing neural models in place of symbolic interpreters.

A3.5.4 Aligning Parallel and Overlapping Expressions

Content summaries can facilitate comparison and knowledge integration in the absence of full structural alignment. In the limiting case of a *complete* structural mismatch between QNR expressions, their summary embeddings can still be compared. To the extent that high-level structures partially align, comparison can proceed based on matching to some limited depth. At points of structural divergence, comparison can fall back on summaries: Where subexpressions differ, their summaries (whether cached or constructed) can be compared; likewise, a subexpression summary in one expression can be compared to a lexical embedding in the other. Appendix A1 discusses how matches can be rejected or applied through soft unification.

Upgrading the expressive power of lexical-level embeddings can facilitate structural alignment by shifting burdens of semantic expressiveness away from syntax: Condensing simple expressions into embeddings avoids a potential source of irregularity, the sequential order of lexical-level elements need not be used to encode emphasis or semantic priority, and the semantic differences between active and passive voice need not be encoded through differences in syntax and grammar. Accordingly, similar meanings become easier to express in parallel syntactic forms.

If expressions with similar semantic content—representing similar things, properties, relationships, roles—are cast in a parallel syntactic form, they become easier to compare. Regularizing structure need not sacrifice expressive capacity: Expression-level nuances that in NL are expressed through alternative syntactic forms can quite generally be represented by embeddings that modify expression-level meaning.

1. While blurring the distinction between values and variables; see Section A1.4.3, which notes potential relationships between constraint-based unification and variable binding in logic programming.

Thus, structural regularization, enabled by expressive embeddings and explicit graphs, can facilitate structural alignment and semantic comparison of related expressions. In addition, however, structural regularization can facilitate transformations among alternative canonical forms, potentially facilitating translation between representational dialects in heterogeneous NL$^+$ corpora. Regularization need not adhere to a uniform standard.

A4 Compositional Lexical Units

Embeddings with explicit compositional structure may offer advantages in efficient learning and generalization.

Section 7.1.3 noted that the properties of vector addition can enable semantic compositionality without recourse to syntax; the present discussion examines the potential role of *explicit* forms of compositionality in learning and representation. Among the considerations are:

- Efficiently representing large vocabularies
- Parallels to natural language vocabularies
- Parallels to NLP input encodings
- Inductive bias toward efficient generalization

The usual disclaimer applies: The aim here is neither to predict nor prescribe particular representations, but to explore what amounts to a lower bound on potential representational capabilities. Explicit vector compositionality, would, however, require explicit architectural support.

A4.1 Motivation and Basic Approach

Because few neural models write and read large stores of neurally encoded information, prospects for building large QNR corpora raise novel questions of storage and practicality. Section A5.5 outlines an approach (using dictionaries of composable vector components) that can be compact, efficient, and expressive. The present discussion considers how and why explicit compositionality within vector representations may be a natural choice for reasons other than efficiency.

A key intuition is that sets of *lexical components* (like morphemes in natural languages) can be composed to represent distinct *lexical units* (like words and phrases that represent objects, actions, classes, relationships, functions, *etc.*[1]), and that *composite lexical units* can best be regarded and implemented as single vectors in QNRs. For concreteness, the discussion here will assume that lexical-component embeddings are concatenated to form lexical-unit embeddings,[2] then melded by shallow feed-forward transformations to form unified representations.

A key underlying assumption is that discrete vocabularies are useful, whether to encode embeddings compactly (Appendix A5), or to provide an inductive bias toward compositional representations. (Note that compact encodings can combine discrete vectors with continuous scalars to designate points on continuous manifolds; see Section A5.5.4).

A4.2 Efficiently Representing Vast Vocabularies

The on-board memories of GPUs and TPUs can readily store $>10^7$ embeddings for fast access.[3] This capacity is orders of magnitude beyond the number of English words, yet using these embeddings as *components* of lexical units can provide much more.

If sets of potential lexical-unit embeddings are Cartesian products of sets of lexical-component embeddings, then potential vocabularies are enormous. Cartesian-product spaces in which (for example) 2 to 4 components are drawn from 10^7 options would offer 10^{14} to 10^{28} potential lexical-unit embeddings; of these, one can expect that a tiny fraction—yet an enormous number—would be potentially useful in describing the world. To represent expressions as strings (Appendix A5), 3 bytes of key information per lexical component would be ample.

A4.3 Parallels to Natural Language Vocabularies

Lexical units in NL vocabularies are commonly built of multiple morphemes, including roots, affixes,[4] and words embedded in compounds or multiword units.[5]

1. As already noted, this use of "lexical units" abuses standard terminology in linguistics.

2. Concatenation can be modeled as addition of blockwise-sparse vectors, and addition of dense vectors would arguably be superior. However, using addition in place of concatenation would (in application) increase storage costs by a small factor, and would (at present) incur a substantial explanatory cost.

3. See Section A5.5.5.

4. English builds on >1300 roots and affixes (prefixsuffix.com 2008).

5. Here used in the standard linguistic sense (also termed "lexical items").

If we view NL as a model for potential NL$^+$ constructs, then it is natural to consider analogues of morphemes in embedding spaces, and to seek lexical-level semantic compositionality through explicit composition of building blocks in which the meanings of components are, as in NL, a joint result of their combination. This approach can make lexical components themselves targets of learning and thereby expand the scope of useful, accessible vocabulary.

Medical terminology illustrates the role of lexical-level compositionality in building a language adapted to a rich domain.[1] Most medical terms are built of sequences of parts ("cardio+vascular") or words ("primary visual cortex"). Wikipedia (2021) lists 510 word parts (prefixes, roots, and suffixes) used in medical terminology, while a large medical dictionary defines ~125,000 distinct, often multi-word terms (Dorland 2007), a number that approaches an estimate (~200,000) of the number of words in the English language as a whole.[2]

Refining and expanding the store of applicable lexical components from hundreds or thousands to millions or more would greatly increase the potential semantic resolution of medical language at a lexical level. Medicine, of course, occupies only a corner of a semantic universe that embraces many fields and extends far beyond what our words can readily describe.

A4.4 Parallels to NLP Input Encodings

There are substantial parallels and contrasts between input encodings in current NLP and compositional embeddings in potential QNR processing systems Table A4.1):

- In the proposed mode of QNR processing, inputs are lexical components concatenated to form lexical units; in Transformer-based NL processing, inputs are words and subwords extracted from strings through tokenization.
- In the QNR case, a very large vocabulary of lexical components is composed to form a vast Cartesian-product space of lexical units; in the NLP case, a smaller vocabulary of lexical units is built from word fragments and common words (in BERT, ~30,000 "wordpieces").

1. A vocabulary which describes structures, functions, relationships, processes, observations, evidence, interventions and causality in systems of extraordinary complexity and human importance.

2. A count that omits, for example, inflected forms (Brysbaert *et al.* 2016).

- In the QNR case, the composition of lexical units is determined by representation learning; In the NLP case, the decomposition of strings is determined by a tokenization algorithm.
- NLP models dynamically infer (and attempt to disambiguate) lexical-level representations from tokenized text; in QNR processing, input embeddings are explicit lexical-level products of previous representation learning. Thus, lexical-level QNR inputs are roughly comparable to hidden-layer representations in an NLP model.

Table A4.1: Input representations used in current NLP and prospective QNR processing.

	Typical NLP models	**Proposed QNR models**
Input units:	wordpiece tokens	component embeddings
Vocabulary size:	~10^4–10^5	~10^7–10^{28}
Embedding origins:	learned representations	learned representations
Initial processing:	multiple attention layers	MoE blending layer[1]

A4.5 Inductive Bias Toward Efficient Generalization

The ability to represent specific lexical units as compositions of more general semantic components could potentially support both systematic generalization and efficient learning, including improved sample efficiency. An important lexical component will typically occur far more frequently than the lexical units that contain it, and learning about a component can provide knowledge regarding lexical units that have not yet been encountered.[2] Indeed, without the inductive bias provided by composition, it might be difficult to learn truly large vocabularies that form well-structured semantic spaces.

As an NL illustration of this principle, consider "primary visual cortex" again: A reader who knows little or nothing of neuroanatomy will know the general meanings of "primary" and "visual" and "cortex", having encountered

1. MoE = mixture of experts (see Fedus, Zoph, and Shazeer 2021).

2. In addition to these considerations, note that components could potentially occupy relatively simple and well-structured semantic spaces, facilitating their interpretation even in the absence of specific training examples. Improving the interpretability of novel lexical components would feed through to improvements in the interpretability of novel elements of a lexical-unit vocabulary.

these terms in diverse contexts. With this knowledge, one can understand that "primary visual cortex" is likely to mean something like "the part of the brain that first processes visual information", even if this term has never before been seen. A more refined understanding can build on this.

This familiar principle carries over to the world of potential QNR representations, where exploitation of compositional lexical-level semantics promises to support learning with broad scope and effective generalization.

A4.6 A Note on Discretized Embeddings

In typical applications, reading is far more frequent than writing, hence mapping continuous internal representations to discrete external representations need not be computationally efficient. This output task can be viewed as either translation or vector quantification. Lexical components that are distant from existing embeddings may represent discoveries worth recording in an expanded vocabulary. Because components populate continuous vector spaces, discretization is compatible with differentiable representation learning of components and their semantics.

A5 Compact QNR Encodings

String representations of QNRs, in conjunction with discretized vector spaces and graph-construction operators, can provide compact and efficient QNR encodings.

"Premature optimization is the root of all evil." — Donald Knuth[1]

NL$^+$ expressions can be implemented compactly by combining operator-based representations of graph structures with extensible dictionaries of discretized embeddings; the latter provide mechanisms for what can be regarded as lossy compression, but can also be regarded as providing a useful inductive bias (Section A4.5). The content of QNR corpora can be represented as byte

1. Knuth (1974). Because compression is (at most) a downstream research priority, Knuth's warning against premature optimization is relevant and suggests that the value of this appendix is questionable. There is, however, good reason to *explore the scope* for efficient, scalable implementations: A sketch of future options can help to free exploratory research from premature efficiency concerns—or worse, a reluctance to consider applications at scale.

strings approximately as compact as NL text by exploiting key–value stores of embeddings and graph-construction operators.[1] The memory footprint these stores need not strain the low-latency memory resources of current machines.

The purpose of this appendix is not to argue for a particular approach, but to show that a potential challenge—the scale of QNR storage footprints—can be met in at least one practical way.

Note that the considerations here have nothing to do with neural computation *per se*, but are instead in the domains of algorithm and data-structure design (often drawing on programming language implementation concepts, *e.g.*, environments and variable binding). From a neural computation perspective, the mechanisms must by design be transparent, which is to say, invisible.

A5.1 Levels of Representational Structure

Prospective QNR repositories include representational elements at three levels of scale:

- ***Embeddings:*** vector attributes at a level comparable to words
- ***Expressions:*** graphs at a level comparable to sentences and paragraphs
- ***References:*** graph links at the level of citations and document structures

In brief, expressions are graphs that bear vector attributes and can include reference-links to other expression-level graphs. There is no important semantic distinction between expression-level and larger-scale graph structures; the key considerations involve interactions between scale, anticipated patterns of use, and implementation efficiency.

A formal notation would distinguish between QNRs as mathematical objects (graph and attributes), QNRs as computational objects (inference-time data structures that represent graphs and attributes), and encodings that represent and evaluate to QNR objects in a computational environment. The following discussion relies on context to clarify meaning.

1. The set of mechanisms outlined here is intended to be illustrative rather than exhaustive, detailed, or optimal. The discussion touches on tutorial topics for the sake of readers who may notice computational puzzles without immediately recognizing their solutions.

A5.2 Explicit Graph Objects *vs.* String Encodings

A relatively simple computational implementation of QNRs would represent embeddings as unshared numerical vectors,[1] and graphs as explicit data structures.[2] Repositories and active computations would share this direct, bulky representational scheme.

Natural language expressions are more compact: NL words in text strings are far smaller than high-dimensional vector embeddings, and graph structures are implicit in NL syntax, which requires no pointer-like links.

Inference-time representations of NL^+ expressions may well be bulky, but so are the inference-time representations of NL expressions in neural NLP. And as with NL, stored QNR expressions can be represented compactly as byte strings.

A5.3 Compact Expression Strings

A QNR corpus can be represented as a key–value store that contains embeddings (numerical vectors), operators (executable code), and encoded QNR expressions (strings that are parsed into keys and internal references). In this scheme, expressions encoded as byte-strings evaluate to expression-level QNR graphs that can include references that define graphs at the scale of documents and corpora.

In more detail:

- ***Keys*** designate operators, embeddings, or expression-strings in a key–value store.
- ***Expression-strings*** are byte strings[3] that are parsed into keys and indices.
- ***Indices*** designate subexpressions in a string.
- ***Operators*** are graph-valued functions[4] of fixed arity that act on sequences of subexpressions (operands).
- ***Subexpressions*** are keys, indices or operator-operand sequences.
- ***Embeddings*** are graph attributes.

1. "Unshared" in the sense that each attribute-slot would designate a distinct, potentially unique vector object.

2. "Explicit" in the sense that each arc would be represented by a pointer-like reference.

3. Potentially bit strings.

4. An extended scheme (Section A5.5.4) allows vector-valued operators with vector and scalar operands.

- ***Reference*** is shorthand for "expression-string key" (typically a stopping point in lazy evaluation).

A5.3.1 From Strings to Syntax Graphs

Parsing an expression-string is straightforward: Operators are functions of fixed arity, the initial bytes of an expression-string designate an operator, and adherence to a standard prefix notation enables parsing of the rest. A first level of decoding yields a graph in which operator nodes link to child nodes that correspond to embeddings, operators, and references to other expressions.[1]

Thus, expression-strings represent graphs in which most intra-expression links are implicit in the composition of graph-valued operators and their products,[2] while the overhead of designating explicit, inter-expression links (several bytes per reference-key) is in effect amortized over the linked expressions. Accordingly, QNR graphs on a scale comparable to NL syntactic structures need not incur per-link, pointer-like overhead.

A5.3.2 Lazy Evaluation

Lazy evaluation enables the piecemeal decoding and expansion of QNR graphs that are too large to fully decompress. To support lazy evaluation, references to expression-strings can evaluate to graph objects (presented as a vector of nodes), or can be left unevaluated. A key–value store then can support lazy evaluation by returning either an expression-string or, when available, the corresponding graph object; a server that retrieves objects for processing can query an expression-store with a key and invoke evaluation if the store returns a string. This mechanism also supports the construction of shared and cyclic graphs.

A5.4 Graph-Construction Operators

Parsing associates each construction operator with a sequence of operands that can be evaluated to produce (or provide a decoded access to) embeddings, references, and root nodes of graph-objects. Simple construction operators can treat their arguments as atomic and opaque; more powerful operators can access the contents of evaluated graph-valued operands or the contents of the operator's evaluation-time context.

1. Internal, index-encoded links require a bit of additional bookkeeping (*i.e.*, remembering subexpression positions) but can describe DAGs and cyclic graphs.

2. Indices (~1 byte?) account for the rest.

Graph structure can be specified explicitly by using a tree notation in conjunction with a representation (*e.g.*, node labels and references) that can distinguish and reference nodes to construct cyclic and convergent paths. Graph construction operators can augment these explicit mechanisms by factoring and abstracting common graph patterns (*e.g.*, the equivalent of common sentence structures); conditional procedural operators can do more. Comprehensive sets of operators ("graphpieces"?) need not greatly burden storage capacity.

A5.4.1 Operators Can Combine Opaque Arguments

A simple class of operators would return a copy of a graph-template—an arbitrarily structured "graph patch"—in which variables are instantiated with arguments that are treated as opaque, atomic tokens. In this approach, the products of operator composition are constrained: Links between graph-patches can target root-nodes returned by operators.[1]

A5.4.2 Paths Can Select Graph Components

A path through a connected graph can designate the position of any node relative to any other, provided that links can be distinguished in navigating from node to node.[2] Path-based access can be implemented by operators that have access to decoded graph contexts.

A5.5 Vocabularies of Embeddings

The computational costs of processing NL$^+$ expressions will depend in part on the storage footprint of their vector embeddings, which in turn will depend on the sizes both of vocabularies and of the vectors themselves.

A5.5.1 Basic Architectural Considerations

Current neural language models tokenize text strings and map a modest vocabulary of tokens[3] to a corresponding vocabulary of embeddings. Processing

1. Or other nodes if links include an index into a returned node-vector. Note that distinguished root nodes are artifacts of encoding that need not be semantically visible. Similar remarks apply to references that target expression-strings.

2. *E.g.*, sequences of car and cdr operators can navigate arbitrary graphs of Lisp cons cells. Paths have no fixed size and could be represented by strings in a key–value store.

3. *E.g.*, ~2^{16} byte pairs or ~30,000 word pieces (Devlin *et al.* 2019).

QNRs encoded as expression-strings requires a similar mapping of tokens (keys) to lexical-component embeddings, but prospective vocabularies are orders of magnitude larger. The use of large vocabularies mandates the use of scalable key–value stores; with this choice, vocabulary size affects neither neural model size nor computational cost.

The use of low-latency storage for frequently used embedding vectors could, however, impose substantial burdens, with requirements scaling as vocabulary size × vector dimensionality × numerical precision.[1] Baseline values for these parameters can place storage concerns in a quantitative context.

A5.5.2 Vocabulary Size

In a continuous vector space, "vocabulary size" becomes meaningful only through vector quantization, the use of identical vectors in multiple contexts.[2] Discrete vectors can be compared to NL words or word parts, and vectors produced by concatenating quantized vectors can be compared to words built of combinations of parts.

A5.5.3 Lexical Component Embeddings

Potential QNR representations and applications (Section 11) are sufficiently diverse (some far from "linguistic" as ordinarily understood) that it is difficult to generalize about the appropriate granularity of vector quantization or the extent to which it should be employed. Different applications will call for different trade-offs between compression and performance, and quantization has been found to improve rather than degrade neural representation learning in some domains.[3]

More can be said about vocabulary size, however, in the context of NL^+ representations that constitute (merely) strong generalizations of NL. To provide a round-number baseline value, consider an NL^+ representation that employs a vocabulary of *distinct lexical components*[4] that is 50 times larger than the vocabulary of *distinct words* in the English language:[5] For present

1. Values of other kinds need not impose similar overheads: Sets of operators need not be large, while expression strings can be retrieved from lower-cost, higher-latency stores.

2. In the literature, the use of low-precision numerical representations is also termed "vector quantization".

3. Agustsson *et al.* (2017), Oord, Vinyals, and Kavukcuoglu (2018), Kaiser and Bengio (2018), Razavi, Oord, and Vinyals (2019), Łańcucki *et al.* (2020), and Zhao *et al.* (2021)

4. Roughly corresponding to morphemes in NL.

5. If lexical units are typically *compositions* of lexical components (Appendix A4), then the size of the potential encoded vocabulary is far larger.

purposes, ~200,000 words is a reasonable estimate of the latter,[1] implying a baseline NL$^+$ vocabulary of 10 million lexical-component embedding vectors.

A5.5.4 Composing Components

As discussed in Appendix A4, a richer vocabulary of lexical units can be constructed by composition, for example, by concatenating lexical-component embeddings. This generative mechanism parallels what we see in natural languages, where many (even most) words are composed of various word-parts, TAM-C affixes, or other words.

Discrete composition: Expression-strings can readily describe composite lexical units: Vector-valued operators with vector arguments can denote concatenations of any number of components. Considering only discrete combinations, operators that accept 2 to 4 arguments from a vocabulary of 10^7 embeddings can define product vocabularies of 10^{14} to 10^{28} composites. These are candidates for use as lexical units, and even tiny useful fractions correspond to vast vocabularies. Appendix A4 explores composite embedding representations from perspectives that include efficiency in learning and semantics in use.

Weighted combination: Some entities are drawn from distributions characterized by continuous properties that include size, color, age, and probability. Discrete vocabularies are a poor fit to continuous semantics, but operators that accept numerical arguments can specify weighted combinations of vectors, and hence continuous manifolds in semantic spaces.

For example, a key that designates a discrete vector embedding $\boldsymbol{a}$ could be replaced by keys designating a vector-valued operator f_i, a scalar parameter w, and embeddings $\boldsymbol{c}$ and $\boldsymbol{d}$, for example, a linear combination:

$$f_i(w, \boldsymbol{c}, \boldsymbol{d}) = w\boldsymbol{c} + (1 - w)\boldsymbol{d}.$$

A suitable range of operators can generalize this scheme to nonlinear functions and higher-dimensional manifolds.

A5.5.5 Dimensionality, Numerical Precision, and Storage Requirements

Transformer-based models typically map sequences of tokens to sequences of high-dimensional embeddings (in the baseline version of BERT, 768 dimen-

1. Vocabulary size depends on how "distinct words" are defined; reasonable definitions and methodologies yield numbers that differ, but not by orders of magnitude. See Brysbaert *et al.* (2016).

sions). Though high dimensionality is perhaps necessary for hidden states that are used to both represent *and support processing* of rich, non-local semantic information, one might expect that word-level semantic units could be represented by embeddings of lower dimensionality. This is empirically true: Shrinking input (word-level) embeddings by more than an order of magnitude (from 768 to 64 dimensions) results in a negligible loss of performance (Lan *et al.* 2020). In estimating potential storage requirements, 100 (but not 10 or 1000) dimensions seems like a reasonable baseline value for representations of broadly word-like lexical components. Forming lexical units by concatenation of components (desirable on several grounds; see Appendix A4) would yield larger input embeddings without increasing storage requirements.

BERT models are typically trained with 32-bit precision, but for use at inference time, parameters throughout the model can be reduced to 8-bit (Prato, Charlaix, and Rezagholizadeh 2020), ternary (Wei Zhang *et al.* 2020), and even binary (H. Bai *et al.* 2020) precision with little loss of performance. As a baseline, allowing 8-bit precision for embeddings at the input interface of a QNR-based inference system seems generous.

When combined with the baseline vocabulary size suggested above (10^7 component embeddings), these numbers suggest a baseline scale for an NL$^+$ key–value store:

$$\begin{aligned}\text{Storage} &= \text{vocabulary size} \times \text{dimensionality} \times \text{precision}\\ &\approx 10{,}000{,}000 \text{ elements} \times 100 \text{ dimensions} \times 1 \text{ bytes}\\ &\approx 1 \text{ GB}.\end{aligned}$$

For comparison, current GPUs and TPUs typically provide on-board memory >10 GB, and are integrated into systems that provide terabytes of RAM.

Given the expected power-law (Zif-like) distribution of use frequencies, implementations in which small local stores of embeddings are backed by large remote stores would presumably experience overwhelmingly local memory traffic.[1] In many tasks (*e.g.*, fetching content used to answer human queries), occasional millisecond-range latencies would presumably be acceptable.

1. As an illustrative NL example, in its first 10^7 words, the Google Books corpus contains about 10^4 distinct words (by a generous definition that includes misspellings), a ratio of one new word in 10^3, while in its first 10^9 words, it contains about 10^5 distinct words, a ratio of one in 10^4. See Brysbaert *et al.* (2016).

A5.5.6 A Note on Non-Lexical Embeddings

Prospective, fully elaborated NL$^+$ systems would employ more than just lexical-level embeddings; for example, embeddings that in some sense summarize expressions can enable shallow processing (skimming), or can serve as keys for near-neighbor retrieval that implements semantic associative memory over large corpora.[1] The costs of summaries can in some sense be amortized over the expressions they summarize.

Like natural language, NL$^+$ expressions can be represented by compact byte strings. Stores of discrete embeddings can support large (and through composition, vast) vocabularies of differentiable semantic representations within an acceptable footprint for low-latency memory; in other words, NL$^+$ corpora can be represented approximately as efficiently and compactly as corpora of NL text. Neither necessity nor optimality is claimed for the approach outlined here.

1. To support near-neighbor indexing, embeddings should be unique, not elements of a "vocabulary".

References

Abramson, Josh, Arun Ahuja, Iain Barr, Arthur Brussee, Federico Carnevale, Mary Cassin, Rachita Chhaparia, *et al.* 2021. "Imitating Interactive Intelligence," arXiv: 2012.05672 [cs.LG].

Abu-Salih, Bilal, Marwan Al-Tawil, Ibrahim Aljarah, Hossam Faris, Pornpit Wongthongtham, Kit Yan Chan, and Amin Beheshti. 2021. "Relational Learning Analysis of Social Politics using Knowledge Graph Embedding." *Data Mining and Knowledge Discovery* 35 (4): 1497–1536.

Addanki, Ravichandra, Peter W. Battaglia, David Budden, Andreea Deac, Jonathan Godwin, Thomas Keck, Wai Lok Sibon Li, *et al.* 2021. "Large-scale graph representation learning with very deep GNNs and self-supervision," arXiv: 2107.09422 [cs.LG].

Adi, Yossi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. "Fine-grained Analysis of Sentence Embeddings Using Auxiliary Prediction Tasks," arXiv: 1608.04207 [cs.CL].

Agustsson, Eirikur, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc V Gool. 2017. "Soft-to-Hard Vector Quantization for End-to-End Learning Compressible Representations." In *Advances in Neural Information Processing Systems*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, vol. 30. Curran Associates, Inc.

Akoury, Nader, Kalpesh Krishna, and Mohit Iyyer. 2019. "Syntactically Supervised Transformers for Faster Neural Machine Translation." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics,* 1269–1281. Florence, Italy: Association for Computational Linguistics.

Ali, Mehdi, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Sahand Sharifzadeh, Volker Tresp, and Jens Lehmann. 2021. "PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings." *Journal of Machine Learning Research* 22 (82): 1–6.

Allamanis, Miltiadis, Marc Brockschmidt, and Mahmoud Khademi. 2018. "Learning to Represent Programs with Graphs," arXiv: 1711.00740 [cs.LG].

Allamanis, Miltiadis, Pankajan Chanthirasegaran, Pushmeet Kohli, and Charles Sutton. 2017. "Learning Continuous Semantic Representations of Symbolic Expressions." In *Proceedings of the 34th International Conference on Machine Learning*, edited by Doina Precup and Yee Whye Teh, 70:80–88. Proceedings of Machine Learning Research. PMLR.

Allan, Rutger J. 2013. "Exploring Modality's Semantic Space Grammaticalization, Subjectification and the case of $o\phi\epsilon\iota\lambda\omega$." *Glotta* 89:1–46.

Allen, Carl, and Timothy Hospedales. 2019. "Analogies Explained: Towards Understanding Word Embeddings." In *Proceedings of the 36th International Conference on Machine Learning*, edited by Kamalika Chaudhuri and Ruslan Salakhutdinov, 97:223–231. Proceedings of Machine Learning Research. PMLR.

Alon, Uri, Meital Zilberstein, Omer Levy, and Eran Yahav. 2019. "Code2vec: Learning Distributed Representations of Code." *Proc. ACM Program. Lang.* (New York, NY, USA) 3 (POPL).

Amiridze, Nino, and Temur Kutsia. 2018. *Anti-Unification and Natural Language Processing.* EasyChair Preprint no. 203.

Arabshahi, Forough, Jennifer Lee, Mikayla Gawarecki, Kathryn Mazaitis, Amos Azaria, and Tom Mitchell. 2021. "Conversational Neuro-Symbolic Commonsense Reasoning," arXiv: 2006.10022 `[cs.AI]`.

Arivazhagan, Naveen, Ankur Bapna, Orhan Firat, Roee Aharoni, Melvin Johnson, and Wolfgang Macherey. 2019. "The Missing Ingredient in Zero-Shot Neural Machine Translation," arXiv: 1903.07091 `[cs.CL]`.

arXiv. 2021. "arXiv Monthly Submissions." Accessed January 16, 2021. https://arxiv.org/stats/monthly_submissions.

Bai, Haoli, Wei Zhang, Lu Hou, Lifeng Shang, Jing Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. 2020. "BinaryBERT: Pushing the Limit of BERT Quantization," arXiv: 2012.15701 `[cs.CL]`.

Bai, Yunsheng, Hao Ding, Ken Gu, Yizhou Sun, and Wei Wang. 2020. "Learning-Based Efficient Graph Similarity Computation via Multi-Scale Convolutional Set Matching." *Proceedings of the AAAI Conference on Artificial Intelligence* 34 (04): 3219–3226.

Banino, Andrea, Adrià Puigdomènech Badia, Raphael Köster, Martin J. Chadwick, Vinicius Zambaldi, Demis Hassabis, Caswell Barry, Matthew Botvinick, Dharshan Kumaran, and Charles Blundell. 2020. "MEMO: A Deep Network for Flexible Combination of Episodic Memories," arXiv: 2001.10913 `[cs.LG]`.

Banino, Andrea, Jan Balaguer, and Charles Blundell. 2021. "PonderNet: Learning to Ponder," arXiv: 2107.05407 `[cs.LG]`.

Bansal, Kshitij, Sarah Loos, Markus Rabe, Christian Szegedy, and Stewart Wilcox. 2019. "HOList: An Environment for Machine Learning of Higher Order Logic Theorem Proving." In *Proceedings of the 36th International Conference on Machine Learning*, edited by Kamalika Chaudhuri and Ruslan Salakhutdinov, 97:454–463. Proceedings of Machine Learning Research. PMLR.

Baroni, Marco, and Roberto Zamparelli. 2010. "Nouns are Vectors, Adjectives are Matrices: Representing Adjective-Noun Constructions in Semantic Space." In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 1183–1193. Cambridge, MA: Association for Computational Linguistics.

Battaglia, Peter W., Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, *et al.* 2018. "Relational inductive biases, deep learning, and graph networks," arXiv: 1806.01261 `[cs.LG]`.

Bauer, Lisa, Yicheng Wang, and Mohit Bansal. 2018. "Commonsense for Generative Multi-Hop Question Answering Tasks." In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4220–4230. Brussels, Belgium: Association for Computational Linguistics.

Bear, Daniel, Chaofei Fan, Damian Mrowca, Yunzhu Li, Seth Alter, Aran Nayebi, Jeremy Schwartz, *et al.* 2020. "Learning Physical Graph Representations from Visual Scenes." In *Advances in Neural Information Processing Systems*, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:6027–6039. Curran Associates, Inc.

Belohlavek, Radim. 2011. "What is a Fuzzy Concept Lattice? II." In *Proceedings of the 13th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing*, 19–26. RSFDGrC'11. Moscow, Russia: Springer-Verlag.

Beltagy, Iz, Matthew E. Peters, and Arman Cohan. 2020. "Longformer: The Long-Document Transformer," arXiv: 2004.05150 [cs.CL].

Benhamou, Frédéric. 1995. "Interval constraint logic programming." In *Constraint Programming: Basics and Trends,* edited by Andreas Podelski, 1–21. Berlin, Heidelberg: Springer Berlin Heidelberg.

Benhamou, Frédéric, and William J. Older. 1997. "Applying interval arithmetic to real, integer, and boolean constraints." *The Journal of Logic Programming* 32 (1): 1–24.

Blacoe, William, and Mirella Lapata. 2012. "A Comparison of Vector-Based Representations for Semantic Composition." In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning,* 546–556. EMNLP-CoNLL '12. Jeju Island, Korea: Association for Computational Linguistics.

Bobrow, Daniel G., and Terry Winograd. 1977. "An Overview of KRL, a Knowledge Representation Language." *Cognitive Science* 1 (1): 3–46. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1207/s15516709cog0101_2.

Bommasani, Rishi, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, *et al.* 2021. "On the Opportunities and Risks of Foundation Models," arXiv: 2108.07258 [cs.LG].

Borsley, Robert, and Kersti Börjars. 2011. *Non-transformational syntax: Formal and explicit models of grammar.* John Wiley & Sons.

Bostrom, Nick. 2014. *Superintelligence: Paths, Dangers, Strategies.* New York: Oxford University Press.

Botha, Jan A., Zifei Shan, and Daniel Gillick. 2020. "Entity Linking in 100 Languages." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 7833–7845. Online: Association for Computational Linguistics.

Botne, Robert, and Tiffany L. Kershner. 2008. "Tense and cognitive space: On the organization of tense/aspect systems in Bantu languages and beyond." 19 (2): 145–218.

Bousmalis, Konstantinos, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. "Domain Separation Networks." In *Advances in Neural Information Processing Systems,* edited by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, vol. 29. Curran Associates, Inc.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, *et al.* 2020. "Language Models are Few-Shot Learners." In *Advances in Neural Information Processing Systems,* edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:1877–1901. Curran Associates, Inc.

Brundage, Miles, Shahar Avin, Jack Clark, Helen Toner, Peter Eckersley, Ben Garfinkel, Allan Dafoe, *et al.* 2018. "The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation," arXiv: 1802.07228 [cs.AI].

Brysbaert, Marc, Michaël Stevens, Paweł Mandera, and Emmanuel Keuleers. 2016. "How Many Words Do We Know? Practical Estimates of Vocabulary Size Dependent on Word Definition, the Degree of Language Input and the Participant's Age." *Frontiers in Psychology* 7:1116.

Cai, HongYun, Vincent W. Zheng, and Kevin Chen-Chuan Chang. 2018. "A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications." *IEEE Transactions on Knowledge and Data Engineering* 30 (9): 1616–1637.

Cai, Ruichu, Zijian Li, Pengfei Wei, Jie Qiao, Kun Zhang, and Zhifeng Hao. 2019. "Learning Disentangled Semantic Representation for Domain Adaptation." In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19,* 2060–2066. International Joint Conferences on Artificial Intelligence Organization, July.

Camacho, Alberto, and Sheila A. McIlraith. 2019. "Towards Neural-Guided Program Synthesis for Linear Temporal Logic Specifications," arXiv: 1912.13430 [cs.AI].

Campero, Andres, Aldo Pareja, Tim Klinger, Josh Tenenbaum, and Sebastian Riedel. 2018. "Logical Rule Induction and Theory Learning Using Neural Theorem Proving," arXiv: 1809.02193 [cs.AI].

Cao, Nicola De, and Thomas Kipf. 2018. "MolGAN: An implicit generative model for small molecular graphs," arXiv: 1805.11973 [stat.ML].

Cao, Yixin, Zhiyuan Liu, Chengjiang Li, Zhiyuan Liu, Juanzi Li, and Tat-Seng Chua. 2019. "Multi-Channel Graph Neural Network for Entity Alignment." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 1452–1461. Florence, Italy: Association for Computational Linguistics.

Caplan, D, and G S Waters. 1999. "Verbal Working Memory and Sentence Comprehension." *The Behavioral and Brain Sciences* 22 (1): 77–94.

Cappart, Quentin, Didier Chételat, Elias Khalil, Andrea Lodi, Christopher Morris, and Petar Veličković. 2021. "Combinatorial optimization and reasoning with graph neural networks," arXiv: 2102.09544 `[cs.LG]`.

Carion, Nicolas, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. "End-to-End Object Detection with Transformers." In *Computer Vision – ECCV 2020*, edited by Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, 213–229. Cham: Springer International Publishing.

Carlini, Nicholas, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, *et al.* 2021. "Extracting Training Data from Large Language Models," arXiv: 2012.07805 `[cs.CR]`.

Cases, Ignacio, Clemens Rosenbaum, Matthew Riemer, Atticus Geiger, Tim Klinger, Alex Tamkin, Olivia Li, *et al.* 2019. "Recursive Routing Networks: Learning to Compose Modules for Language Understanding." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 3631–3648. Minneapolis, Minnesota: Association for Computational Linguistics.

Chen, Benson, Regina Barzilay, and Tommi Jaakkola. 2019. "Path-Augmented Graph Transformer Network," arXiv: 1905.12712 `[cs.LG]`.

Chen, Dawn, Joshua C. Peterson, and Thomas L. Griffiths. 2017. "Evaluating vector-space models of analogy," arXiv: 1705.04416 `[cs.CL]`.

Chen, Mark, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, *et al.* 2021. "Evaluating Large Language Models Trained on Code," arXiv: 2107.03374 `[cs.LG]`.

Chen, Wei, and Mark Fuge. 2019. "Synthesizing Designs With Interpart Dependencies Using Hierarchical Generative Adversarial Networks." *Journal of Mechanical Design* 141 (11).

Cimiano, Philipp, Andreas Hotho, and Steffen Staab. 2005. "Learning Concept Hierarchies from Text Corpora Using Formal Concept Analysis." *J. Artif. Int. Res.* (El Segundo, CA, USA) 24 (1): 305–339.

Cingillioglu, Nuri, and Alessandra Russo. 2020. "Learning Invariants through Soft Unification." In *Advances in Neural Information Processing Systems,* edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:8186–8197. Curran Associates, Inc.

Clark, Peter, Oyvind Tafjord, and Kyle Richardson. 2020. "Transformers as Soft Reasoners over Language," arXiv: 2002.05867 `[cs.CL]`.

Clark, Stephen, Alexander Lerchner, Tamara von Glehn, Olivier Tieleman, Richard Tanburn, Misha Dashevskiy, and Matko Bosnjak. 2021. "Formalising Concepts as Grounded Abstractions," arXiv: 2101.05125 `[cs.AI]`.

Cohan, Arman, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. "SPECTER: Document-level Representation Learning using Citation-informed Transformers." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics,* 2270–2282. Online: Association for Computational Linguistics.

Conneau, Alexis, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. "What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties." In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 2126–2136. Melbourne, Australia: Association for Computational Linguistics.

Constant, Mathieu, Gülşen Eryiğit, Johanna Monti, Lonneke van der Plas, Carlos Ramisch, Michael Rosner, and Amalia Todirascu. 2017. "Multiword Expression Processing: A Survey." *Comput. Linguist.* (Cambridge, MA, USA) 43 (4): 837–892.

Cummins, Chris, Hugh Leather, Zacharias Fisches, Tal Ben-Nun, Torsten Hoefler, and Michael O'Boyle. 2020. "Deep Data Flow Analysis," arXiv: 2012.01470 `[cs.PL]`.

Currey, Anna, and Kenneth Heafield. 2019. "Incorporating Source Syntax into Transformer-Based Neural Machine Translation." In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers),* 24–33. Florence, Italy: Association for Computational Linguistics.

Das, Abhishek, Satwik Kottur, José M. F. Moura, Stefan Lee, and Dhruv Batra. 2017. "Learning Cooperative Visual Dialog Agents with Deep Reinforcement Learning." In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2970–2979.

Desai, Karan, and Justin Johnson. 2021. "VirTex: Learning Visual Representations from Textual Annotations," arXiv: 2006.06666 `[cs.CV]`.

Deshpande, Shrinath, and Anurag Purwar. 2019. "Computational Creativity Via Assisted Variational Synthesis of Mechanisms Using Deep Generative Models." *Journal of Mechanical Design* 141 (12).

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.

Distill Team. 2021. "About Distill: Dedicated to clear explanations of machine learning." Distill. Accessed January 16, 2021. https://distill.pub/about/.

Dorland, William Alexander Newman. 2007. *Dorland's Illustrated Medical Dictionary.* Dorland's Medical Dictionary Series. Saunders Elsevier.

Drexler, K. Eric. 2019. *Reframing Superintelligence: Comprehensive AI Services as General Intelligence.* Technical Report 2019-1. Future of Humanity Institute, University of Oxford. https://www.fhi.ox.ac.uk/reframing/.

Edel, Yves, E M Rains, and N J A Sloane. 2002. "On Kissing Numbers in Dimensions 32 to 128," arXiv: 0207291 `[math.CO]`.

Eppe, Manfred, Ewen Maclean, Roberto Confalonieri, Oliver Kutz, Marco Schorlemmer, Enric Plaza, and Kai-Uwe Kühnberger. 2018. "A computational framework for conceptual blending." *Artificial Intelligence* 256:105–129.

Eslami, S. M. Ali, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S. Morcos, Marta Garnelo, Avraham Ruderman, *et al.* 2018. "Neural scene representation and rendering." *Science* 360 (6394): 1204–1210. eprint: https://science.sciencemag.org/content/360/6394/1204.full.pdf.

Ethayarajh, Kawin, David Duvenaud, and Graeme Hirst. 2019. "Towards Understanding Linear Word Analogies." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics,* 3253–3262. Florence, Italy: Association for Computational Linguistics.

Fan, Angela, Claire Gardent, Chloé Braud, and Antoine Bordes. 2021. "Augmenting Transformers with KNN-Based Composite Memory for Dialog." *Transactions of the Association for Computational Linguistics* 9:82–99.

Fedus, William, Barret Zoph, and Noam Shazeer. 2021. "Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity," arXiv: 2101.03961 `[cs.LG]`.

Felty, Amy, and Dale Miller. 1988. "Specifying theorem provers in a higher-order logic programming language." In *9th International Conference on Automated Deduction,* edited by Ewing Lusk and Ross Overbeek, 61–80. Berlin, Heidelberg: Springer Berlin Heidelberg.

Feng, Zhangyin, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, *et al.* 2020. "CodeBERT: A Pre-Trained Model for Programming and Natural Languages." In *Findings of the Association for Computational Linguistics: EMNLP 2020,* 1536–1547. Online: Association for Computational Linguistics.

Ferreira, Deborah, and André Freitas. 2020. "Premise Selection in Natural Language Mathematical Texts." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics,* 7365–7374. Online: Association for Computational Linguistics.

Févry, Thibault, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. "Entities as Experts: Sparse Memory Access with Entity Supervision." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 4937–4951. Online: Association for Computational Linguistics.

Fey, Matthias, Jan E. Lenssen, Christopher Morris, Jonathan Masci, and Nils M. Kriege. 2020. "Deep Graph Matching Consensus," arXiv: 2001.09621 `[cs.LG]`.

Fu, Cong, Chao Xiang, Changxu Wang, and Deng Cai. 2018. "Fast Approximate Nearest Neighbor Search With The Navigating Spreading-out Graph," arXiv: 1707.00143 `[cs.LG]`.

Galassi, Andrea, Kristian Kersting, Marco Lippi, Xiaoting Shao, and Paolo Torroni. 2020. “Neural-Symbolic Argumentation Mining: An Argument in Favor of Deep Learning and Reasoning.” *Frontiers in Big Data* 2:52.

Ganguly, Soumyajit, and Vikram Pudi. 2017. “Paper2vec: Combining Graph and Text Information for Scientific Paper Representation.” In *Advances in Information Retrieval*, edited by Joemon M Jose, Claudia Hauff, Ismail Sengor Altıngovde, Dawei Song, Dyaa Albakour, Stuart Watt, and John Tait, 383–395. Cham: Springer International Publishing.

Ganin, Yaroslav, Sergey Bartunov, Yujia Li, Ethan Keller, and Stefano Saliceti. 2021. “Computer-Aided Design as Language,” arXiv: 2105.02769 `[cs.CV]`.

Ganin, Yaroslav, and Victor Lempitsky. 2015. “Unsupervised Domain Adaptation by Backpropagation.” In *Proceedings of the 32nd International Conference on Machine Learning*, edited by Francis Bach and David Blei, 37:1180–1189. Proceedings of Machine Learning Research. Lille, France: PMLR.

Ganter, Bernhard, and Rudolf Wille. 1997. “Applied Lattice Theory: Formal Concept Analysis.” In *In General Lattice Theory, G. Grätzer editor, Birkhäuser.* Preprints.

———. 1999. *Formal Concept Analysis: Mathematical Foundations.* 1st ed. Translated by C. Franzke. Springer-Verlag Berlin Heidelberg.

Garcez, Artur d’Avila, and Luis C. Lamb. 2020. “Neurosymbolic AI: The 3rd Wave,” arXiv: 2012.05876 `[cs.AI]`.

Gärdenfors, Peter. 2000. *Conceptual Spaces: The Geometry of Thought.* The MIT Press.

Gentner, Dedre, and Kenneth D. Forbus. 2011. “Computational models of analogy.” *WIREs Cognitive Science* 2 (3): 266–276. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcs.105.

Giaquinto, Marcus. 2015. “The Epistemology of Visual Thinking in Mathematics.” In *Stanford Encyclopedia of Philosophy*, edited by Edward N. Zalta. https://plato.stanford.edu/archives/spr2020/entries/epistemology-visual-thinking/.

Gilmer, Justin, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. "Neural Message Passing for Quantum Chemistry." In *Proceedings of the 34th International Conference on Machine Learning - Volume 70,* 1263–1272. ICML'17. Sydney, NSW, Australia: JMLR.org.

Google. 2020. "About Google." Accessed December 28, 2020. https://about.google/.

Goyal, Anirudh, and Yoshua Bengio. 2021. "Inductive Biases for Deep Learning of Higher-Level Cognition," arXiv: 2011.15091 `[cs.LG]`.

Goyal, Anirudh, Aniket Didolkar, Nan Rosemary Ke, Charles Blundell, Philippe Beaudoin, Nicolas Heess, Michael Mozer, and Yoshua Bengio. 2021. "Neural Production Systems," arXiv: 2103.01937 `[cs.AI]`.

Grohe, Martin. 2020. "Word2vec, Node2vec, Graph2vec, X2vec: Towards a Theory of Vector Embeddings of Structured Data." In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems,* 1–16. PODS'20. Portland, OR, USA: Association for Computing Machinery.

Guarino, Nicola. 2009. "The Ontological Level: Revisiting 30 Years of Knowledge Representation." In *Conceptual Modeling: Foundations and Applications: Essays in Honor of John Mylopoulos,* edited by Alexander T. Borgida, Vinay K. Chaudhri, Paolo Giorgini, and Eric S. Yu, 52–67. Berlin, Heidelberg: Springer Berlin Heidelberg.

Guhe, Markus, Alison Pease, Alan Smaill, Martin Schmidt, Helmar Gust, Kai-Uwe Kühnberger, and Ulf Krumnack. 2010. "Mathematical reasoning with higher-order anti-unifcation" [in English]. In *Proceedings of the 32nd Annual Conference of the Cognitive Science Society,* 1992–1997.

Gulcehre, Caglar, Sarath Chandar, Kyunghyun Cho, and Yoshua Bengio. 2018. "Dynamic Neural Turing Machine with Continuous and Discrete Addressing Schemes." *Neural Comput.* (Cambridge, MA, USA) 30 (4): 857–884.

Guo, Daya, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, *et al.* 2021. "GraphCodeBERT: Pre-training Code Representations with Data Flow," arXiv: 2009.08366 `[cs.SE]`.

Guu, Kelvin, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. "REALM: Retrieval-Augmented Language Model Pre-Training," arXiv: 2002.08909 `[cs.CL]`.

Han, Yizeng, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. 2021. “Dynamic Neural Networks: A Survey,” arXiv: 2102.04906 [cs.CV].

Härkönen, Erik, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. 2020. “GANSpace: Discovering Interpretable GAN Controls,” arXiv: 2004.02546 [cs.CV].

Hashimoto, Tatsunori B., David Alvarez-Melis, and Tommi S. Jaakkola. 2016. “Word Embeddings as Metric Recovery in Semantic Spaces.” *Transactions of the Association for Computational Linguistics* 4:273–286.

He, Xin, Kaiyong Zhao, and Xiaowen Chu. 2021. “AutoML: A survey of the state-of-the-art.” *Knowledge-Based Systems* 212:106622.

Heimann, Mark, Haoming Shen, Tara Safavi, and Danai Koutra. 2018. “REGAL: Representation Learning-Based Graph Alignment.” In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 117–126. CIKM ’18. Torino, Italy: Association for Computing Machinery.

Hernandez, Danny, and Tom B. Brown. 2020. “Measuring the Algorithmic Efficiency of Neural Networks,” arXiv: 2005.04305 [cs.LG].

Hinton, Geoffrey. 2021. “How to represent part-whole hierarchies in a neural network,” arXiv: 2102.12627 [cs.CV].

Hofstadter, Douglas. 2009. “Analogy as the Core of Cognition.” Presidential Lecture given at Stanford University, Stanford, CA, September 10, 2009. https://shc.stanford.edu/multimedia/analogy-core-cognition.

Hogan, Aidan, Eva Blomqvist, Michael Cochez, Claudia d’Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, *et al.* 2021. “Knowledge Graphs,” arXiv: 2003.02320 [cs.AI].

Hossain, MD. Zakir, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. 2019. “A Comprehensive Survey of Deep Learning for Image Captioning.” *ACM Comput. Surv.* (New York, NY, USA) 51 (6).

Howard, Phil. 2021. “Computational Propaganda Research Project.” Accessed January 16, 2021. https://demtech.oii.ox.ac.uk/.

Huang, James Y., Kuan-Hao Huang, and Kai-Wei Chang. 2021. "Disentangling Semantics and Syntax in Sentence Embeddings with Pre-trained Language Models." In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1372–1379. Online: Association for Computational Linguistics.

Ivanova, Anna A., Shashank Srikant, Yotaro Sueoka, Hope H. Kean, Riva Dhamala, Una-May O'Reilly, Marina U. Bers, and Evelina Fedorenko. 2020. "Comprehension of computer code relies primarily on domain-general executive resources." *bioRxiv*, eprint: https://www.biorxiv.org/content/early/2020/04/18/2020.04.16.045732.full.pdf.

Jaffar, Joxan, and Michael J. Maher. 1994. "Constraint logic programming: a survey." Special Issue: Ten Years of Logic Programming, *The Journal of Logic Programming* 19-20:503–581.

Jaradeh, Mohamad Yaser, Allard Oelen, Kheir Eddine Farfar, Manuel Prinz, Jennifer D'Souza, Gábor Kismihók, Markus Stocker, and Sören Auer. 2019. "Open Research Knowledge Graph: Next Generation Infrastructure for Semantic Scholarly Knowledge." In *Proceedings of the 10th International Conference on Knowledge Capture*, 243–246. K-CAP '19. Marina Del Rey, CA, USA: Association for Computing Machinery.

Jaulin, Luc. 2006. "Computing Minimal-Volume Credible Sets Using Interval Analysis; Application to Bayesian Estimation." *IEEE Transactions on Signal Processing* 54 (9): 3632–3636.

Jayaram Subramanya, Suhas, Fnu Devvrit, Harsha Vardhan Simhadri, Ravishankar Krishnawamy, and Rohan Kadekodi. 2019. "DiskANN: Fast Accurate Billion-point Nearest Neighbor Search on a Single Node." In *Advances in Neural Information Processing Systems*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, vol. 32. Curran Associates, Inc.

Ji, Shaoxiong, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2021. "A Survey on Knowledge Graphs: Representation, Acquisition, and Applications." *IEEE Transactions on Neural Networks and Learning Systems*, 1–21.

Jiang, Ming, Jennifer D'Souza, Sören Auer, and J. Stephen Downie. 2020. "Improving Scholarly Knowledge Representation: Evaluating BERT-Based Models for Scientific Relation Classification." In *Digital Libraries at Times of Massive Societal Transition,* edited by Emi Ishita, Natalie Lee San Pang, and Lihong Zhou, 3–19. Cham: Springer International Publishing.

Jiang, YiDing, Shixiang (Shane) Gu, Kevin P Murphy, and Chelsea Finn. 2019. "Language as an Abstraction for Hierarchical Deep Reinforcement Learning." In *Advances in Neural Information Processing Systems,* edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, vol. 32. Curran Associates, Inc.

Johnson, Jeff, Matthijs Douze, and Hervé Jégou. 2019. "Billion-scale similarity search with GPUs." *IEEE Transactions on Big Data,* 1–1.

Joshi, Mandar, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. "SpanBERT: Improving Pre-training by Representing and Predicting Spans." *Transactions of the Association for Computational Linguistics* 8:64–77.

Kahneman, Daniel. 1961. "An analytical model of the semantic differential," University of California, Berkeley.

———. 2011. *Thinking, fast and slow.* New York: Farrar, Straus / Giroux.

Kahng, Andrew B. 2018. "Machine Learning Applications in Physical Design: Recent Results and Directions." In *Proceedings of the 2018 International Symposium on Physical Design,* 68–73. ISPD '18. Monterey, California, USA: Association for Computing Machinery.

Kaiser, Łukasz, and Samy Bengio. 2018. "Discrete Autoencoders for Sequence Models," arXiv: 1801.09797 [cs.LG].

Katharopoulos, Angelos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. "Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention," arXiv: 2006.16236 [cs.LG].

Kato, Akihiko, Hiroyuki Shindo, and Yuji Matsumoto. 2016. "Construction of an English Dependency Corpus incorporating Compound Function Words." In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16),* 1667–1671. Portorož, Slovenia: European Language Resources Association (ELRA).

Kazemi, Seyed Mehran, and David Poole. 2018. “SimplE Embedding for Link Prediction in Knowledge Graphs.” In *Advances in Neural Information Processing Systems,* edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, vol. 31. Curran Associates, Inc.

Kazi, Anees, Luca Cosmo, Nassir Navab, and Michael Bronstein. 2020. “Differentiable Graph Module (DGM) for Graph Convolutional Networks,” arXiv: 2002.04999 `[cs.LG]`.

Kehagias, Ath. 2011. “Some remarks on the lattice of fuzzy intervals.” Special Issue on Information Engineering Applications Based on Lattices, *Information Sciences* 181 (10): 1863–1873.

Keskar, Nitish Shirish, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. “CTRL: A Conditional Transformer Language Model for Controllable Generation,” arXiv: 1909.05858 `[cs.CL]`.

Khandelwal, Urvashi, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. “Generalization through Memorization: Nearest Neighbor Language Models,” arXiv: 1911.00172 `[cs.CL]`.

Kim, Young Jin, and Hany Hassan. 2020. “FastFormers: Highly Efficient Transformer Models for Natural Language Understanding.” In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing,* 149–158. Online: Association for Computational Linguistics.

Kingma, Diederik P., and Prafulla Dhariwal. 2018. “Glow: Generative Flow with Invertible 1x1 Convolutions,” arXiv: 1807.03039 `[stat.ML]`.

Klys, Jack, Jake Snell, and Richard Zemel. 2018. “Learning Latent Subspaces in Variational Autoencoders.” In *Advances in Neural Information Processing Systems,* edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, vol. 31. Curran Associates, Inc.

Knight, Kevin. 1989. “Unification: A Multidisciplinary Survey.” *ACM Comput. Surv.* (New York, NY, USA) 21 (1): 93–124.

Knuth, Donald. 1974. *The Art of Computer Programming.* Addison-Wesley.

Koncel-Kedziorski, Rik, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. "Text Generation from Knowledge Graphs with Graph Transformers." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers),* 2284–2293. Minneapolis, Minnesota: Association for Computational Linguistics.

Kucik, Andrzej Stanisław, and Konstantin Korovin. 2018. "Premise selection with neural networks and distributed representation of features," arXiv: 1807.10268 `[cs.AI]`.

Lamb, Luís C., Artur d'Avila Garcez, Marco Gori, Marcelo O.R. Prates, Pedro H.C. Avelar, and Moshe Y. Vardi. 2020. "Graph Neural Networks Meet Neural-Symbolic Computing: A Survey and Perspective." In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20,* edited by Christian Bessiere, 4877–4884. Survey track. International Joint Conferences on Artificial Intelligence Organization, July.

Lample, Guillaume, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Herve Jegou. 2019. "Large Memory Layers with Product Keys." In *Advances in Neural Information Processing Systems,* edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, vol. 32. Curran Associates, Inc.

Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations," arXiv: 1909.11942 `[cs.CL]`.

Łańcucki, Adrian, Jan Chorowski, Guillaume Sanchez, Ricard Marxer, Nanxin Chen, Hans J. G. A. Dolfing, Sameer Khurana, Tanel Alumäe, and Antoine Laurent. 2020. "Robust Training of Vector Quantized Bottleneck Models," arXiv: 2005.08520 `[cs.LG]`.

Larsen, Anders Boesen Lindbo, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. 2016. "Autoencoding beyond Pixels Using a Learned Similarity Metric." In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48,* 1558–1566. ICML'16. New York, NY, USA: JMLR.org.

Lawrence, John, and Chris Reed. 2020. "Argument Mining: A Survey." *Computational Linguistics* 45 (4): 765–818.

Lazaridou, Angeliki, and Marco Baroni. 2020. "Emergent Multi-Agent Communication in the Deep Learning Era," arXiv: 2006.02419 [cs.CL].

Lazaridou, Angeliki, Alexander Peysakhovich, and Marco Baroni. 2017. "Multi-Agent Cooperation and the Emergence of (Natural) Language," arXiv: 1612.07182 [cs.CL].

Le, Hung, Truyen Tran, and Svetha Venkatesh. 2020. "Neural Stored-program Memory." In *International Conference on Learning Representations.*

Lee, Chelsea. 2019. "Welcome, singular "they"." *APA Style* (blog), October 31, 2019. https://apastyle.apa.org/blog/singular-They.

Lee, Haimin. 2019. "15 years of Google Books." *Google Blog* (blog), October 17, 2019. https://www.blog.google/products/search/15-years-google-books/.

Lee, Kuang-Huei, Hamid Palangi, Xi Chen, Houdong Hu, and Jianfeng Gao. 2019. "Learning Visual Relation Priors for Image-Text Matching and Image Captioning with Neural Scene Graph Generators," arXiv: 1909.09953 [cs.CV].

Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, *et al.* 2020. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks." In *Advances in Neural Information Processing Systems,* edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:9459–9474. Curran Associates, Inc.

Lewis, Richard L., Shravan Vasishth, and Julie A. Van Dyke. 2006. "Computational principles of working memory in sentence comprehension." *Trends in Cognitive Sciences* 10 (10): 447–454.

Li, Ruiyu, Makarand Tapaswi, Renjie Liao, Jiaya Jia, Raquel Urtasun, and Sanja Fidler. 2017. "Situation Recognition with Graph Neural Networks," arXiv: 1708.04320 [cs.CV].

Li, Yujia, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. 2019. "Graph Matching Networks for Learning the Similarity of Graph Structured Objects." In *Proceedings of the 36th International Conference on Machine Learning,* edited by Kamalika Chaudhuri and Ruslan Salakhutdinov, 97:3835–3845. Proceedings of Machine Learning Research. PMLR.

Li, Yujia, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter Battaglia. 2018. "Learning Deep Generative Models of Graphs," arXiv: 1803.03324 [cs.LG].

Liao, Haiguang, Wentai Zhang, Xuliang Dong, Barnabas Poczos, Kenji Shimada, and Levent Burak Kara. 2019. "A Deep Reinforcement Learning Approach for Global Routing." *Journal of Mechanical Design* 142 (6).

Lieto, Antonio, Antonio Chella, and Marcello Frixione. 2017. "Conceptual Spaces for Cognitive Architectures: A lingua franca for different levels of representation." *Biologically Inspired Cognitive Architectures* 19:1–9.

Liu, Qi, Matt J. Kusner, and Phil Blunsom. 2020. "A Survey on Contextual Embeddings," arXiv: 2003.07278 [cs.CL].

Liu, Rui, Junjie Hu, Wei Wei, Zi Yang, and Eric Nyberg. 2017. "Structural Embedding of Syntactic Trees for Machine Comprehension." In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 815–824. Copenhagen, Denmark: Association for Computational Linguistics.

Liu, Rui, Yu Liu, Xinyu Gong, Xiaogang Wang, and Hongsheng Li. 2019. "Conditional Adversarial Generative Flow for Controllable Image Synthesis." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

Liu, Shusen, Peer-Timo Bremer, Jayaraman J. Thiagarajan, Vivek Srikumar, Bei Wang, Yarden Livnat, and Valerio Pascucci. 2018. "Visual Exploration of Semantic Relationships in Neural Word Embeddings." *IEEE Transactions on Visualization and Computer Graphics* 24 (1): 553–562.

Liu, Xiaodong, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. "Multi-Task Deep Neural Networks for Natural Language Understanding." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4487–4496. Florence, Italy: Association for Computational Linguistics.

Liu, Yinhan, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. "Multilingual Denoising Pre-training for Neural Machine Translation." *Transactions of the Association for Computational Linguistics* 8:726–742.

Lu, Jiasen, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. "ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks." In *Advances in Neural Information Processing Systems,* edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, vol. 32. Curran Associates, Inc.

Lu, Kevin, Aditya Grover, Pieter Abbeel, and Igor Mordatch. 2021. "Pretrained Transformers as Universal Computation Engines," arXiv: `2103.05247 [cs.LG]`.

Lu, Yichao, Phillip Keung, Faisal Ladhak, Vikas Bhardwaj, Shaonan Zhang, and Jason Sun. 2018. "A neural interlingua for multilingual machine translation." In *Proceedings of the Third Conference on Machine Translation: Research Papers,* 84–92. Brussels, Belgium: Association for Computational Linguistics.

Luketina, Jelena, Nantas Nardelli, Gregory Farquhar, Jakob Foerster, Jacob Andreas, Edward Grefenstette, Shimon Whiteson, and Tim Rocktäschel. 2019. "A Survey of Reinforcement Learning Informed by Natural Language." In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19,* 6309–6317. International Joint Conferences on Artificial Intelligence Organization, July.

Mager, Manuel, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. "GPT-too: A Language-Model-First Approach for AMR-to-Text Generation." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics,* 1846–1852. Online: Association for Computational Linguistics.

Malekmohamadi, Soroor, Faramarz Safi-Esfahani, and Morteza Karimian-kelishadrokhi. 2020. "A Review on Neural Turing Machine (NTM)." *SN Computer Science* 1 (6): 333.

Malinowski, Mateusz, Carl Doersch, Adam Santoro, and Peter Battaglia. 2018. "Learning Visual Question Answering by Bootstrapping Hard Attention." In *Proceedings of the European Conference on Computer Vision (ECCV).*

Martin, W. A., and R. J. Fateman. 1971. "The MACSYMA System." In *Proceedings of the Second ACM Symposium on Symbolic and Algebraic Manipulation,* 59–75. SYMSAC '71. Los Angeles, California, USA: Association for Computing Machinery.

Martinez, M., A. M. H. Abdel-Fattah, U. Krumnack, D. Gómez-Ramírez, A. Smaill, T. R. Besold, A. Pease, M. Schmidt, M. Guhe, and K.-U. Kühnberger. 2017. "Theory blending: extended algorithmic aspects and examples." *Annals of Mathematics and Artificial Intelligence* 80 (1): 65–89.

Martino, Giovanni Da San, Stefano Cresci, Alberto Barrón-Cedeño, Seunghak Yu, Roberto Di Pietro, and Preslav Nakov. 2020. "A Survey on Computational Propaganda Detection." In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20,* edited by Christian Bessiere, 4826–4832. Survey track. International Joint Conferences on Artificial Intelligence Organization, July.

McCann, Bryan, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. "The Natural Language Decathlon: Multitask Learning as Question Answering," arXiv: 1806.08730 `[cs.CL]`.

McShane, Marjorie, and Sergei Nirenburg. 2012. "A Knowledge Representation Language for Natural Language Processing, Simulation and Reasoning." *International Journal of Semantic Computing* 06 (01): 3–23. eprint: https://doi.org/10.1142/S1793351X12400016.

Medina, Jesús, Manuel Ojeda-Aciego, and Peter Vojtáš. 2004. "Similarity-based unification: a multi-adjoint approach." Selected Papers from EUSFLAT 2001, *Fuzzy Sets and Systems* 146 (1): 43–62.

Menezes, Telmo, and Camille Roth. 2021. "Semantic Hypergraphs," arXiv: 1908.10784 `[cs.IR]`.

Messick, Samuel J. 1957. "Metric Properties of the Semantic Differential." *Educational and Psychological Measurement* 17 (2): 200–206.

Miech, Antoine, Jean-Baptiste Alayrac, Ivan Laptev, Josef Sivic, and Andrew Zisserman. 2021. "Thinking Fast and Slow: Efficient Text-to-Visual Retrieval with Transformers," arXiv: 2103.16553 `[cs.CV]`.

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig. 2013. "Linguistic Regularities in Continuous Space Word Representations." In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* 746–751. Atlanta, Georgia: Association for Computational Linguistics.

Min, Sewon, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2020. "Knowledge Guided Text Retrieval and Reading for Open Domain Question Answering," arXiv: 1911.03868 `[cs.CL]`.

Minervini, Pasquale, Matko Bosnjak, Tim Rocktäschel, and Sebastian Riedel. 2018. "Towards Neural Theorem Proving at Scale," arXiv: 1807.08204 [cs.AI].

Minervini, Pasquale, Matko Bošnjak, Tim Rocktäschel, Sebastian Riedel, and Edward Grefenstette. 2019. "Differentiable Reasoning on Large Knowledge Bases and Natural Language," arXiv: 1912.10824 [cs.LG].

———. 2020. "Differentiable reasoning on large knowledge bases and natural language." In *Proceedings of the AAAI conference on artificial intelligence,* 34:5182–5190. 04.

Mitra, Bhaskar, and Nick Craswell. 2018. "An Introduction to Neural Information Retrieval." *Foundations and Trends® in Information Retrieval* 13 (1): 1–126.

Mo, Kaichun, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy J. Mitra, and Leonidas J. Guibas. 2019. "StructureNet: Hierarchical Graph Networks for 3D Shape Generation." *ACM Trans. Graph.* (New York, NY, USA) 38 (6).

Moens, Marie-Francine. 2018. "Argumentation mining: How can a machine acquire common sense and world knowledge?" *Argument & Computation* 9 (1): 1–14.

Mordatch, Igor, and Pieter Abbeel. 2018. "Emergence of Grounded Compositional Language in Multi-Agent Populations," arXiv: 1703.04908 [cs.AI].

Moses, Joel. 2012. "Macsyma: A personal history." *Journal of Symbolic Computation* 47 (2): 123–130.

Munroe, Randall. 2012. "Words for Small Sets." *XKCD* (blog), June 20, 2012. https://xkcd.com/1070/.

Narayanan, Annamalai, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. 2017. "graph2vec: Learning Distributed Representations of Graphs," arXiv: 1707.05005 [cs.AI].

Nguyen, Xuan-Phi, Shafiq Joty, Steven C. H. Hoi, and Richard Socher. 2020. "Tree-structured Attention with Hierarchical Accumulation," arXiv: 2002.08046 [cs.LG].

Noble, James, Alex Potanin, Toby Murray, and Mark S. Miller. 2018. "Abstract and Concrete Data Types vs Object Capabilities." In *Principled Software Development: Essays Dedicated to Arnd Poetzsch-Heffter on the Occasion of his 60th Birthday,* edited by Peter Müller and Ina Schaefer, 221–240. Cham: Springer International Publishing.

Noé, Frank, Alexandre Tkatchenko, Klaus-Robert Müller, and Cecilia Clementi. 2020. "Machine Learning for Molecular Simulation." *Annual Review of Physical Chemistry,* 361–390.

Nurse, Derek, and Gérard Philippson. 2006. "Common tense-aspect markers in Bantu." *Journal of African Languages and Linguistics* 27 (2): 155–196.

Odena, Augustus, Kensen Shi, David Bieber, Rishabh Singh, and Charles Sutton. 2020. "BUSTLE: Bottom-up program-Synthesis Through Learning-guided Exploration," arXiv: 2007.14381 `[cs.PL]`.

Oh, Sangeun, Yongsu Jung, Seongsin Kim, Ikjin Lee, and Namwoo Kang. 2019. "Deep Generative Design: Integration of Topology Optimization and Generative Models." *Journal of Mechanical Design* 141 (11).

Older, William, and André Vellino. 1990. "Extending Prolog with Constraint Arithmetic on Real Intervals." In *In Canadian Conference on Computer & Electrical Engineering.* Society Press.

Oord, Aaron van den, Yazhe Li, and Oriol Vinyals. 2019. "Representation Learning with Contrastive Predictive Coding," arXiv: 1807.03748 `[cs.LG]`.

Oord, Aaron van den, Oriol Vinyals, and Koray Kavukcuoglu. 2018. "Neural Discrete Representation Learning," arXiv: 1711.00937 `[cs.LG]`.

OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, *et al.* 2019. "Dota 2 with Large Scale Deep Reinforcement Learning," arXiv: 1912.06680 `[cs.LG]`.

Pan, Shirui, Ruiqi Hu, Sai-Fu Fung, Guodong Long, Jing Jiang, and Chengqi Zhang. 2020. "Learning Graph Embedding With Adversarial Training Methods." *IEEE Transactions on Cybernetics* 50 (6): 2475–2487.

Pan, Shirui, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. 2018. "Adversarially Regularized Graph Autoencoder for Graph Embedding." In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18,* 2609–2615. International Joint Conferences on Artificial Intelligence Organization, July.

Patashnik, Or, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. 2021. "StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery," arXiv: 2103.17249 [cs.CV].

Paulson, Lawrence C. 1986. "Natural deduction as higher-order resolution." *The Journal of Logic Programming* 3 (3): 237–258.

Pearce, Hammond, Baleegh Ahmad, Benjamin Tan, Brendan Dolan-Gavitt, and Ramesh Karri. 2021. *An Empirical Cybersecurity Evaluation of GitHub Copilot's Code Contributions.* arXiv: 2108.09293 [cs.CR].

Peng, Wei, Tuomas Varanka, Abdelrahman Mostafa, Henglin Shi, and Guoying Zhao. 2021. "Hyperbolic Deep Neural Networks: A Survey," arXiv: 2101.04562 [cs.LG].

Polosukhin, Illia, and Alexander Skidanov. 2018. "Neural Program Search: Solving Programming Tasks from Description and Examples," arXiv: 1802.04335 [cs.AI].

Polu, Stanislas, and Ilya Sutskever. 2020. "Generative Language Modeling for Automated Theorem Proving," arXiv: 2009.03393 [cs.LG].

Pólya, G. 1990. *Mathematics and Plausible Reasoning: Induction and analogy in mathematics.* Induction and Analogy in Mathematics. Princeton University Press.

Prato, Gabriele, Ella Charlaix, and Mehdi Rezagholizadeh. 2020. "Fully Quantized Transformer for Machine Translation," arXiv: 1910.10485 [cs.CL].

prefixsuffix.com. 2008. "English Language Roots." Accessed March 17, 2008. http://www.prefixsuffix.com/rootchart.php.

Puri, Ruchir. 2021. "Kickstarting AI for Code: Introducing IBM's Project CodeNet." IBM Research. Accessed May 10, 2021. https://research.ibm.com/blog/codenet-ai-for-code.

Qu, Meng, Jian Tang, and Yoshua Bengio. 2019. "Weakly-supervised Knowledge Graph Alignment with Adversarial Learning," arXiv: 1907.03179 [cs.LG].

Raboh, Moshiko, Roei Herzig, Gal Chechik, Jonathan Berant, and Amir Globerson. 2020. "Differentiable Scene Graphs," arXiv: 1902.10200 [cs.CV].

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, *et al.* 2021. "Learning Transferable Visual Models From Natural Language Supervision," arXiv: 2103.00020 [cs.CV].

Radosavovic, Ilija, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. 2020. "Designing Network Design Spaces," arXiv: 2003.13678 [cs.CV].

Rae, Jack W., Anna Potapenko, Siddhant M. Jayakumar, and Timothy P. Lillicrap. 2019. "Compressive Transformers for Long-Range Sequence Modelling," arXiv: 1911.05507 [cs.LG].

Raghu, Maithra, and Eric Schmidt. 2020. "A Survey of Deep Learning for Scientific Discovery," arXiv: 2003.11755 [cs.LG].

Raina, Ayush, Jonathan Cagan, and Christopher McComb. 2019. "Transferring Design Strategies From Human to Computer and Across Design Problems." *Journal of Mechanical Design* 141 (11).

Raina, Ayush, Christopher McComb, and Jonathan Cagan. 2019. "Learning to Design From Humans: Imitating Human Designers Through Deep Learning." *Journal of Mechanical Design* 141 (11).

Raposo, David, Adam Santoro, David Barrett, Razvan Pascanu, Timothy Lillicrap, and Peter Battaglia. 2017. "Discovering objects and their relations from entangled scene representations," arXiv: 1702.05068 [cs.LG].

Ratner, Alex, Braden Hancock, Jared Dunnmon, Roger Goldman, and Christopher Ré. 2018. "Snorkel MeTaL: Weak Supervision for Multi-Task Learning." In *Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning.* DEEM'18. Houston, TX, USA: Association for Computing Machinery.

Ratner, Alexander, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2020. "Snorkel: rapid training data creation with weak supervision." *The VLDB Journal* 29 (2): 709–730.

Razavi, Ali, Aaron van den Oord, and Oriol Vinyals. 2019. "Generating Diverse High-Fidelity Images with VQ-VAE-2," arXiv: 1906.00446 [cs.LG].

Real, Esteban, Chen Liang, David So, and Quoc Le. 2020. "AutoML-Zero: Evolving Machine Learning Algorithms From Scratch." In *Proceedings of the 37th International Conference on Machine Learning*, edited by Hal Daumé III and Aarti Singh, 119:8007–8019. Proceedings of Machine Learning Research. PMLR.

Ren, Pengzhen, Yun Xiao, Xiaojun Chang, Po-yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. 2021. "A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions." *ACM Comput. Surv.* (New York, NY, USA) 54 (4).

Rezaeinia, Seyed Mahdi, Ali Ghodsi, and Rouhollah Rahmani. 2017. "Improving the Accuracy of Pre-trained Word Embeddings for Sentiment Analysis," arXiv: 1711.08609 `[cs.CL]`.

Ribeiro, Leonardo F. R., Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2020. "Investigating Pretrained Language Models for Graph-to-Text Generation," arXiv: 2007.08426 `[cs.CL]`.

Rocktäschel, Tim, and Sebastian Riedel. 2016. "Learning Knowledge Base Inference with Neural Theorem Provers." In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*, 45–50. San Diego, CA: Association for Computational Linguistics.

———. 2017. "End-to-end Differentiable Proving." In *Advances in Neural Information Processing Systems*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, vol. 30. Curran Associates, Inc.

Rosenbaum, Clemens, Ignacio Cases, Matthew Riemer, and Tim Klinger. 2019. "Routing Networks and the Challenges of Modular and Compositional Computation," arXiv: 1904.12774 `[cs.LG]`.

Russell, Stuart. 2019. *Human compatible: Artificial intelligence and the problem of control.* Penguin.

Sagara, Moriji, Kazuo Yamamoto, Hirohiko Nishimura, and Hiroshi Akuto. 1961. "A Study on the Semantic Structure of Japanese Language by the Semantic Differential Method." *Japanese Psychological Research* 3 (3): 146–156.

Salehi, Amin, and Hasan Davulcu. 2020. "Graph Attention Auto-Encoders." In *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, 989–996.

Sanh, Victor, Thomas Wolf, and Sebastian Ruder. 2018. "A Hierarchical Multi-task Approach for Learning Embeddings from Semantic Tasks," arXiv: 1811.06031 [cs.CL].

Santoro, Adam, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. "Meta-Learning with Memory-Augmented Neural Networks." In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48,* 1842–1850. ICML'16. New York, NY, USA: JMLR.org.

Sarlin, Paul-Edouard, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. 2020. "SuperGlue: Learning Feature Matching With Graph Neural Networks." In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* 4937–4946.

Schlag, Imanol, Paul Smolensky, Roland Fernandez, Nebojsa Jojic, Jürgen Schmidhuber, and Jianfeng Gao. 2020. "Enhancing the Transformer with Explicit Relational Encoding for Math Problem Solving," arXiv: 1910.06611 [cs.LG].

Schramowski, Patrick, Cigdem Turan, Sophie Jentzsch, Constantin Rothkopf, and Kristian Kersting. 2019. "BERT has a Moral Compass: Improvements of ethical and moral values of machines," arXiv: 1912.05238 [cs.CL].

Schwenk, Holger, and Matthijs Douze. 2017. "Learning Joint Multilingual Sentence Representations with Neural Machine Translation." In *Proceedings of the 2nd Workshop on Representation Learning for NLP,* 157–167. Vancouver, Canada: Association for Computational Linguistics.

Sessa, Maria I. 2002. "Approximate reasoning by similarity-based SLD resolution." *Theoretical Computer Science* 275 (1): 389–426.

Shah, Pararth, Marek Fiser, Aleksandra Faust, J. Chase Kew, and Dilek Hakkani-Tur. 2018. "FollowNet: Robot Navigation by Following Natural Language Directions with Deep Reinforcement Learning," arXiv: 1805.06150 [cs.RO].

Shen, Yujun, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. 2020. "Interpreting the Latent Space of GANs for Semantic Face Editing." In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* 9240–9249.

Shi, Jiaxin, Hanwang Zhang, and Juanzi Li. 2019. "Explainable and Explicit Visual Reasoning Over Scene Graphs." In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8368–8376.

Shiv, Vighnesh, and Chris Quirk. 2019. "Novel positional encodings to enable tree-based transformers." In *Advances in Neural Information Processing Systems*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, vol. 32. Curran Associates, Inc.

Simak, Clifford D. 2016. *City (1952).* Gateway.

Simon, Herbert A. 1988. "The Science of Design: Creating the Artificial." *Design Issues* 4 (1/2): 67–82.

Simonovsky, Martin, and Nikos Komodakis. 2018. "GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders," arXiv: 1802.03480 [cs.LG].

Singh, Prem Kumar, C. Aswani Kumar, and Jinhai Li. 2016. "Knowledge Representation Using Interval-Valued Fuzzy Formal Concept Lattice." *Soft Computing* 20 (4): 1485–1502.

Slonim, Noam, Yonatan Bilu, Carlos Alzate, Roy Bar-Haim, Ben Bogin, Francesca Bonin, Leshem Choshen, *et al.* 2021. "An autonomous debating system." *Nature* 591 (7850): 379–384.

Smolka, Gert. 1992. "Feature-constraint logics for unification grammars." *The Journal of Logic Programming* 12 (1): 51–87.

Solaiman, Irene, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, *et al.* 2019. "Release Strategies and the Social Impacts of Language Models," arXiv: 1908.09203 [cs.CL].

Stump, Gary M., Simon W. Miller, Michael A. Yukish, Timothy W. Simpson, and Conrad Tucker. 2019. "Spatial Grammar-Based Recurrent Neural Network for Design Form and Behavior Optimization." *Journal of Mechanical Design* 141 (12).

Sun, Philip. 2020. "Announcing ScaNN: Efficient Vector Similarity Search." *Google AI Blog* (blog), July 28, 2020. https://ai.googleblog.com/2020/07/announcing-scann-efficient-vector.html.

Svyatkovskiy, Alexey, Sebastian Lee, Anna Hadjitofi, Maik Riechert, Juliana Franco, and Miltiadis Allamanis. 2021. "Fast and Memory-Efficient Neural Code Completion." In *The 2021 Mining Software Repositories Conference.* arXiv: 2004.13651 [cs.SE].

Szabó, Zoltán Gendler. 2017. "Compositionality." In *Stanford Encyclopedia of Philosophy*, edited by Edward N Zalta.

Szegedy, Christian. 2020. "A promising path towards autoformalization and general artificial intelligence." In *International Conference on Intelligent Computer Mathematics*, 3–20. Springer.

Tamkin, Alex, Dan Jurafsky, and Noah Goodman. 2020. "Language Through a Prism: A Spectral Approach for Multiscale Language Representations." In *Advances in Neural Information Processing Systems*, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:5492–5504. Curran Associates, Inc.

Tarlow, Daniel, Subhodeep Moitra, Andrew Rice, Zimin Chen, Pierre-Antoine Manzagol, Charles Sutton, and Edward Aftandilian. 2020. "Learning to Fix Build Errors with Graph2Diff Neural Networks." In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, 19–20. ICSEW'20. Seoul, Republic of Korea: Association for Computing Machinery.

Tee, Naeem. 2021. "What's The Average Freelance Writer's Price Per Word?" *Contentfly* (blog), April 9, 2021. https://contentfly.com/blog/whats-the-average-freelance-writers-price-per-word.

Thorne, James, Majid Yazdani, Marzieh Saeidi, Fabrizio Silvestri, Sebastian Riedel, and Alon Halevy. 2020. "Neural Databases," arXiv: 2010.06973 [cs.CL].

Torquato, S., and F. H. Stillinger. 2006. "Exactly solvable disordered sphere-packing model in arbitrary-dimensional Euclidean spaces." *Physical Review E* 73 (3).

Tousch, Anne-Marie, Stéphane Herbin, and Jean-Yves Audibert. 2008. "Semantic Lattices for Multiple Annotation of Images." In *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*, 342–349. MIR '08. Vancouver, British Columbia, Canada: Association for Computing Machinery.

Tran, Chau, Yuqing Tang, Xian Li, and Jiatao Gu. 2020. "Cross-lingual Retrieval for Iterative Self-Supervised Training." In *Advances in Neural Information Processing Systems,* edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:2207–2219. Curran Associates, Inc.

Tripathi, Subarna, Anahita Bhiwandiwalla, Alexei Bastidas, and Hanlin Tang. 2019. "Using Scene Graph Context to Improve Image Generation," arXiv: 1901.03762 [cs.CV].

Trivedi, Rakshit, Jiachen Yang, and Hongyuan Zha. 2020. "GraphOpt: Learning Optimization Models of Graph Formation," arXiv: 2007.03619 [cs.LG].

Turovsky, Barak. 2016. "Ten years of Google Translate." *Google: The Keyword* (blog), April 28, 2016. https://www.blog.google/products/translate/ten-years-of-google-translate.

Van Lierde, H., and Tommy W.S. Chow. 2019. "Query-oriented text summarization based on hypergraph transversals." *Information Processing & Management* 56 (4): 1317–1338.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. "Attention is All You Need." In *Proceedings of the 31st International Conference on Neural Information Processing Systems,* 6000–6010. NIPS'17. Long Beach, California, USA: Curran Associates Inc.

Veličković, Petar, and Charles Blundell. 2021. "Neural Algorithmic Reasoning," arXiv: 2105.02761 [cs.LG].

Veličković, Petar, Lars Buesing, Matthew Overlan, Razvan Pascanu, Oriol Vinyals, and Charles Blundell. 2020. "Pointer Graph Networks." In *Advances in Neural Information Processing Systems,* edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:2232–2244. Curran Associates, Inc.

Velikovich, Leonid, Ian Williams, Justin Scheiner, Petar Aleksic, Pedro Moreno, and Michael Riley. 2018. "Semantic Lattice Processing in Contextual Automatic Speech Recognition for Google Assistant." In *Interspeech 2018,* 2222–2226. https://www.isca-speech.org/archive/Interspeech_2018/pdfs/2453.pdf.

Verga, Pat, Haitian Sun, Livio Baldini Soares, and William W. Cohen. 2020. "Facts as Experts: Adaptable and Interpretable Neural Memory over Symbolic Knowledge," arXiv: 2007.00849 [cs.CL].

Vicente, Agustin. 2018. "Polysemy and word meaning: an account of lexical meaning for different kinds of content words." *Philosophical Studies* 175 (4): 947–968.

Vinyals, Oriol, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, *et al.* 2019. "Grandmaster Level in StarCraft II Using Multi-Agent Reinforcement Learning." *Nature* 575 (7782): 350–354.

Wang, Jizhe, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. "Billion-Scale Commodity Embedding for E-Commerce Recommendation in Alibaba." In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining,* 839–848. KDD '18. London, United Kingdom: Association for Computing Machinery.

Wang, Ke, and Mihai Christodorescu. 2019. "COSET: A Benchmark for Evaluating Neural Program Embeddings," arXiv: 1905.11445 [cs.LG].

Wang, Mingzhe, Yihe Tang, Jian Wang, and Jia Deng. 2017. "Premise Selection for Theorem Proving by Deep Graph Embedding." In *Advances in Neural Information Processing Systems,* edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, vol. 30. Curran Associates, Inc.

Wang, Shenghui, and Rob Koopman. 2017. "Semantic embedding for information retrieval." In *BIR 2017 Workshop on Bibliometric-enhanced Information Retrieval.*

Wang, Wenhan, Sijie Shen, Ge Li, and Zhi Jin. 2020. "Towards Full-line Code Completion with Neural Language Models," arXiv: 2009.08603 [cs.SE].

Wannenwetsch, Anne S., Martin Kiefel, Peter V. Gehler, and Stefan Roth. 2019. "Learning Task-Specific Generalized Convolutions in the Permutohedral Lattice," arXiv: 1909.03677 [cs.CV].

Watters, Nicholas, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. 2017. "Visual Interaction Networks: Learning a Physics Simulator from Video." In *Advances in Neural Information Processing Systems,* edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, vol. 30. Curran Associates, Inc.

Weber, Leon, Pasquale Minervini, Jannes Münchmeyer, Ulf Leser, and Tim Rocktäschel. 2019. "NLProlog: Reasoning with Weak Unification for Question Answering in Natural Language." In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics,* 6151–6161. Florence, Italy: Association for Computational Linguistics.

Wieting, John, Graham Neubig, and Taylor Berg-Kirkpatrick. 2020. "A Bilingual Generative Transformer for Semantic Sentence Embedding." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 1581–1594. Association for Computational Linguistics.

Wikipedia. 2021. *List of medical roots, suffixes and prefixes — Wikipedia, The Free Encyclopedia.* http://en.wikipedia.org/w/index.php?title=List%20of%20medical%20roots%2C%20suffixes%20and%20prefixes&oldid=1029902137. [Online; accessed 23-June-2021].

Wilkins, John. 1668. "An Essay Towards a Real Character, and a Philosophical Language."

Wolchover, N. 2015. *Concerns of an artificial intelligence pioneer. Quanta. April 21.*

Woolley, Sam, and Phil Howard. 2017. *Computational Propaganda Worldwide: Executive Summary.* Working Paper 2017.11. Oxford Internet Institute, University of Oxford. https://www.oii.ox.ac.uk/blog/computational-propaganda-worldwide-executive-summary/.

Wu, Zonghan, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. 2021. "A Comprehensive Survey on Graph Neural Networks." *IEEE transactions on neural networks and learning systems* 32:4–24.

Xu, Peng, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. 2020. "MEGATRON-CNTRL: Controllable Story Generation with External Knowledge Using Large-Scale Language Models." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 2831–2845. Online: Association for Computational Linguistics.

Yang, Guangyu Robert, Madhura R. Joglekar, H. Francis Song, William T. Newsome, and Xiao-Jing Wang. 2019. "Task Representations in Neural Networks Trained to Perform Many Cognitive Tasks." *Nature Neuroscience* 22 (2): 297–306.

Yang, Jianwei, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. 2018. "Graph R-CNN for Scene Graph Generation," arXiv: 1808.00191 `[cs.CV]`.

Yernaux, Gonzague, and Wim Vanhoof. 2019. "Anti-unification in Constraint Logic Programming." *Theory and Practice of Logic Programming* 19 (5-6): 773–789.

Yogatama, Dani, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021. "Adaptive Semiparametric Language Models." *Transactions of the Association for Computational Linguistics* 9:362–373.

You, Jiaxuan, Zhitao Ying, and Jure Leskovec. 2020. "Design Space for Graph Neural Networks." In *Advances in Neural Information Processing Systems,* edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:17009–17021. Curran Associates, Inc.

Yu, Dongfei, Jianlong Fu, Tao Mei, and Yong Rui. 2017. "Multi-level Attention Networks for Visual Question Answering." In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4187–4195.

Yun, Seongjun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. "Graph Transformer Networks." In *Advances in Neural Information Processing Systems,* edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, vol. 32. Curran Associates, Inc.

Zaheer, Manzil, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, *et al.* 2020. "Big Bird: Transformers for Longer Sequences." In *Advances in Neural Information Processing Systems*, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, 33:17283–17297. Curran Associates, Inc.

Zellers, Rowan, Mark Yatskar, Sam Thomson, and Yejin Choi. 2018. "Neural Motifs: Scene Graph Parsing with Global Context." In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5831–5840.

Zhang, Haijun, Shuang Wang, Xiaofei Xu, Tommy W. S. Chow, and Q. M. Jonathan Wu. 2018. "Tree2Vector: Learning a Vectorial Representation for Tree-Structured Data." *IEEE Transactions on Neural Networks and Learning Systems* 29 (11): 5304–5318.

Zhang, Wei, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020. "TernaryBERT: Distillation-aware Ultra-low Bit BERT." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP),* 509–521. Online: Association for Computational Linguistics.

Zhang, Wen, Bibek Paudel, Wei Zhang, Abraham Bernstein, and Huajun Chen. 2019. "Interaction Embeddings for Prediction and Explanation in Knowledge Graphs." In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining,* 96–104. WSDM '19. Melbourne VIC, Australia: Association for Computing Machinery.

Zhang, Ye, Md Mustafizur Rahman, Alex Braylan, Brandon Dang, Heng-Lu Chang, Henna Kim, Quinten McNamara, *et al.* 2017. "Neural Information Retrieval: A Literature Review," arXiv: 1611.06792 [cs.IR].

Zhang, Yian, Alex Warstadt, Haau-Sing Li, and Samuel R. Bowman. 2020. "When Do You Need Billions of Words of Pretraining Data?," arXiv: 2011.04946 [cs.CL].

Zhao, Yang, Ping Yu, Suchismit Mahapatra, Qinliang Su, and Changyou Chen. 2021. "Improve Variational Autoencoder for Text Generation with Discrete Latent Bottleneck," arXiv: 2004.10603 [cs.LG].

Zheng, Cheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. 2020. "Robust Graph Representation Learning via Neural Sparsification." In *Proceedings of the 37th International Conference on Machine Learning,* edited by Hal Daumé III and Aarti Singh, 119:11458–11468. Proceedings of Machine Learning Research. PMLR.

Zhou, Jie, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. "Graph neural networks: A review of methods and applications." *AI Open* 1:57–81.

Zhou, Zhenpeng, Steven Kearnes, Li Li, Richard N. Zare, and Patrick Riley. 2019. "Optimization of Molecules via Deep Reinforcement Learning." *Scientific Reports* 9 (1): 10752.