Banya AI 5

Cache-Dictionary Tokenization and Dynamic Embedding: Near-Free Vocabulary Expansion via a Low-Rank Head

Hyukjin Han (bokkamsun@gmail.com) · ORCID 0009-0007-4132-1707

Banya Research Series, Part 5

Abstract

Tokenization—splitting text into the word units a model handles—is arranged in two layers. The lower layer consists of syllable atoms (Korean phonetic units), and the upper layer is a bundle dictionary that holds frequent words as whole tokens. The vocabulary of the measured model stacks 8000 bundles on top of 2724 syllable atoms, for a total of 10724. Ordinarily, increasing the vocabulary fourfold makes the output head (the final layer that selects the next word) proportionally more expensive in parameters, but this paper makes vocabulary expansion nearly free by giving the output head a low-rank form (approximating the large matrix as a product of two smaller matrices). At vocabulary 10724, channel width 1024, and rank 128, the low-rank head has about 1.5 million parameters, about 7.3 times fewer than the dense head's about 10.98 million. Furthermore, bundle-layer tokens preserve the orthogonal structure of the data plane and the operation plane intact (the operator emergence of Part 3 also appears in bundles, with a direction consistency of 8.1 times). That is, the cache dictionary gains compression without breaking the data-operation structure established by the earlier papers. We also measured that the savings travel together with quality. The normal path, bundles on, predicts holdout text at 5.08 bits per character while encoding the same text with 21.5% fewer tokens; with bundles off, prediction collapses at word-internal positions (11.49 bits per token vs. 6.69 at boundaries), revealing that this system's word knowledge resides in the bundle path.

Measured

The quantitative figures in this paper are actual measurements taken in the writing environment (the frozen elementary-stage model; environment in Appendix B). Values computed directly from the structure, such as the parameter count of the low-rank head, are stated as fact.

Contents

Terminology Legend

Terms are defined in the tables below. One concept goes by one name only, and the text and figures of all seven parts follow these tables. The in-code column gives the identifier that implements the concept in the enclosed code, and the standard-term column gives the closest name in common use; cells are filled only where they differ.

Terminology legend 1. Engine and credit

Term	Meaning	In code	Standard term
exact adjoint	The backward computation derived by hand as closed forms for every operation; the proper name of this engine and its credit chain	`banya_core.py`	manual back-propagation
adjoint	The partner operation of the forward pass; it carries the gradient exactly backward	`bwd` kernels	adjoint
credit	The loss gradient at a hidden state; the responsibility a parameter bears for the outcome	delta (δ)	gradient
credit chain	The skeleton that walks the cache in reverse, applying each adjoint to carry credit back to the input	`backward`	back-propagation
adjoint agreement	The check that the adjoint formulas produce the same values as finite differences	per-part probes	gradient check
circulant-matrix channel mixing	Channel mixing with a circulant matrix, computed as componentwise products under rfft		circular convolution
relative-distance mixing	Mixing positions with shared coefficients that depend only on distance	`m_mat_w_lag`	Toeplitz mixing
temporal mixing	The umbrella name for any path by which other positions' information flows into the current one		token mixing
mixing heads	The number of parallel heads a mixing operation is split into		attention heads
filter ladder	The convolutional channel transform whose filter scale varies with depth, from a few wide filters to many narrow ones	`m_mat_w_filter`	convolutional filterbank
window	The token positions the model sees at once; as a measurement unit, a fixed-length text slice	`T`	context window
probe	The verification script that replays every measured figure together with its target value	`probe` folder	reproduction script
frozen model	A developmental-stage model whose training is stopped and fixed	`model` npz	frozen checkpoint
holdout	Evaluation text never used in training		held-out set

Terminology legend 2. Bi-token and the gate

Term	Meaning	In code	Standard term
bi-token	The structure that gives one token an orthogonal pair of embeddings, a data plane and an operation plane	`USE_BITOKEN`	
data plane	The embedding plane carrying a token's content; it enters at the add slot	`m_mat_w_data_axis`	token embedding
operation plane	The embedding plane carrying a token's operation instruction; it enters only at the multiply slot	`m_mat_w_operate_axis`	
add slot, multiply slot	The residual-addition position where the data plane lands and the gate-multiplication position where the operation plane acts		residual connection, gating
gate	The valve that sets each channel's pass-through between 0 and 1; the previous token's operation plane is injected into it	`m_mat_w_gate`	gating
one-step shift	The one-position displacement by which the previous token's operation plane is injected into the next position's gate		
operation-as-weight	The phenomenon in which operation-plane embeddings emerge as weights without any labels		
consistent operator	An operator that acts in a consistent direction on any data		
direction consistency	The mean cosine of the directions in which one operation bends different data; the quantitative sign of operator emergence	`p_direction_consistency`	
rotation-operator gate	The gate that reads the operation plane as Euler rotation angles; abbreviated in the text as the rotation gate	`paper7_rotation.py`	
primitive operation	An operation the gate can express directly in a single-hop instruction		primitive
depth-excluded discrimination	The deliberately biased measurement that restricts depth to one block so the gate is the only remaining variable	Part 7 probe	
self-feedback	The scoring mode that feeds the model's own output back as input and scores the final state		autoregressive rollout

Terminology legend 3. Rumination and development

Term	Meaning	In code	Standard term
Rumination	Self-organizing learning that pulls embeddings toward their neighbors without computing any gradient	Part 2 probes	self-organization
same-kind group	The set of close-in-meaning concepts gathered from a seed by nearest-cosine traversal		cluster
centroid pull, decay	The two forces of Rumination: pulling toward the mean and restoring toward the original; equilibrium emerges from their balance	`ALPHA`, decay rate	
World-first	The curriculum order that develops sensory, spatial, and logical axes before language	`banya_world_data`	curriculum learning
developmental stage	A checkpoint stage defined by step count and named after human growth		
axis	A property scale formed as a direction inside the embedding: sensory, ordinal, categorical		representation axis

Terminology legend 4. Vocabulary and metacognition

Term	Meaning	In code	Standard term
syllable atom	The syllable-level token forming the bottom layer of the vocabulary	`AtomTokenizer`	character-level token
bundle	A concept token made by binding syllable atoms into the dictionary; used only as a vocabulary unit		subword
cache-dictionary tokenization	The two-layer vocabulary system that puts a bundle dictionary on top of the atom layer	Part 5 probes	contrast with subword tokenization
certain, vague, unknown	The three knowing states separated by the model's own output distribution	pmax, entropy	uncertainty estimation
routing	Assigning an answer's processing path according to the state		routing
reinterpretation	The procedure that re-runs a vague hidden state together with its context to lift it to certain	Part 6 probes	
similarity consolidation	The axis along which concepts are organized by data-plane similarity		
inference traversal	The axis along which inference is carried forward on the operation plane		
self-utterance	The target state that weaves similarity consolidation and inference traversal so the model continues speech by itself		
Banya framework	The fifteen-axiom system forming the series' background; not the ground of the claims in the text		

Symbol Legend

This section defines the symbols used by both the equations and the text of this paper. Local symbols used only within a single equation are given in the symbol line beneath each equation.

Symbol	Name	Definition and meaning	Type
V	vocabulary size	Number of tokens in the model vocabulary; the measured model stacks 8000 bundles on 2724 syllable atoms for a total of 10724	integer
H	channel width	Width of the model's hidden channels; 1024 in the measured model	integer
R	rank	Rank of the low-rank factorization; 128 in the measured model	integer
W_{head}	output head matrix	The large matrix of the final layer that selects the next token; approximated by the product $W_a W_b$ of two smaller matrices	matrix
W_a	first low-rank factor	Front matrix of the low-rank factorization; size $V \times R$	matrix
W_b	second low-rank factor	Rear matrix of the low-rank factorization; size $R \times H$	matrix
$V \times H$	dense head parameter count	Parameter count of the dense output head; the product of vocabulary size and channel width, about 10.98 million	integer
$(V + H) \times R$	low-rank head parameter count	Parameter count of the low-rank output head; the sum of vocabulary and channel multiplied by the rank, about 1.5 million	integer
$\times$	product	Multiplication used to compute parameter counts from dimensions	operator
$\in$	element of	Membership notation stating that a matrix belongs to a real space	operator
$\mathbb{R}$	real numbers	Set symbol stating that matrix entries are real; $\mathbb{R}^{V \times H}$ denotes the space a matrix belongs to	set notation
$\ll$	much less than	The condition $R \ll H$ that the rank is far smaller than the channel width	operator
$\approx$	approximately equal	States that the computed saving ratio is about 7.3	operator
$\pm$	plus-minus	Mean and standard deviation of the per-fragment differences in the table of Section 3.3	operator

1. Introduction

Enlarging the vocabulary increases expressive power but makes the output head expensive. A dense head has as many parameters as the product $V \times H$ of the vocabulary size V and the channel width H, so it grows with the vocabulary. This paper enlarges the vocabulary with a two-layer cache dictionary that stacks a bundle dictionary on top of syllable atoms, and nearly eliminates the cost by factorizing the output head into a low-rank form. We further verify by measurement that bundle-layer tokens preserve the orthogonal structure of the data plane and the operation plane established by the earlier papers. Training uses the engine of Part 1 [2]. The theoretical background lies in the separate axiom document [1], but this paper draws its conclusions from measurement alone, independently of it.

2. Methods

The lower-layer syllable tokenization splits text into phonetic units. The upper-layer bundle dictionary merges frequent words into single tokens by greedy longest match (matching the longest entries first and capturing them whole). Input and output share the same dictionary. The output head approximates the large matrix $W_{\text{head}} \in \mathbb{R}^{V \times H}$ by a product of two smaller matrices $W_a W_b$ ($W_a \in \mathbb{R}^{V \times R}$, $W_b \in \mathbb{R}^{R \times H}$, rank $R \ll H$).

Eq. 2-1. Parameter cost of the low-rank head

$$\text{dense} : V \times H, \qquad \text{low-rank} : (V + H) \times R$$

$$\frac{V \times H}{(V + H) \times R} = \frac{10724 \times 1024}{(10724 + 1024) \times 128} \approx 7.3$$

Formula

Symbols: V is the vocabulary size (10724), H the channel width (1024), and R the rank (128).
How it works: the dense head is as large as the product of vocabulary and channel, whereas the low-rank head is the sum of vocabulary and channel each multiplied only by the rank. If the rank stays fixed, the cost grows only linearly in the vocabulary as it expands.
In practice: at vocabulary 10724 the dense head has about 10.98 million parameters and the low-rank head about 1.5 million, about 7.3 times fewer. Even if the vocabulary grows further, the low-rank head increases only by $V \times R$ and thus grows far more gently.

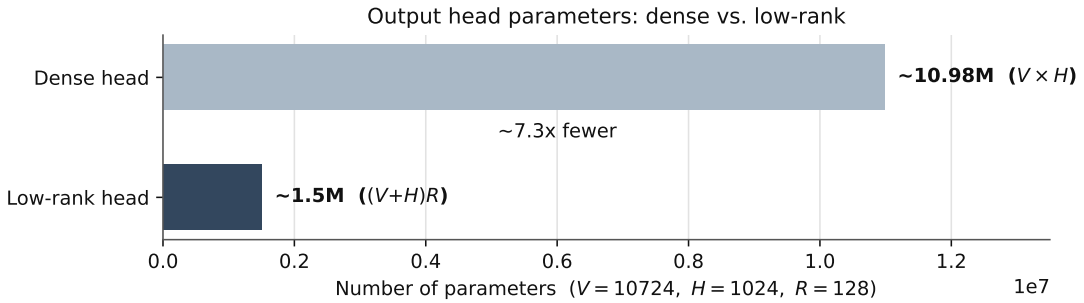
3. Results

3.1 The two-layer cache dictionary and the cost of the low-rank head

Measured

The vocabulary of the measured model stacks 8000 bundles on 2724 syllable atoms, for a total of 10724. At this vocabulary, the low-rank head (rank 128) has about 1.5 million parameters, about 7.3 times fewer than the roughly 10.98 million of a dense head at the same vocabulary (structural computation).

Figure 3-1. Dense head vs. low-rank head at vocabulary 10724 (structural computation)



At the same vocabulary of 10724, the low-rank head produces the output with about 7.3 times fewer parameters than the dense head. Because the low-rank head grows only gently as the vocabulary expands, vocabulary expansion in the two-layer cache dictionary becomes nearly free.

3.2 The bundle layer also preserves the data-operation structure

We measured whether enlarging the vocabulary with bundles breaks the orthogonal structure of the data plane and the operation plane established by the earlier papers.

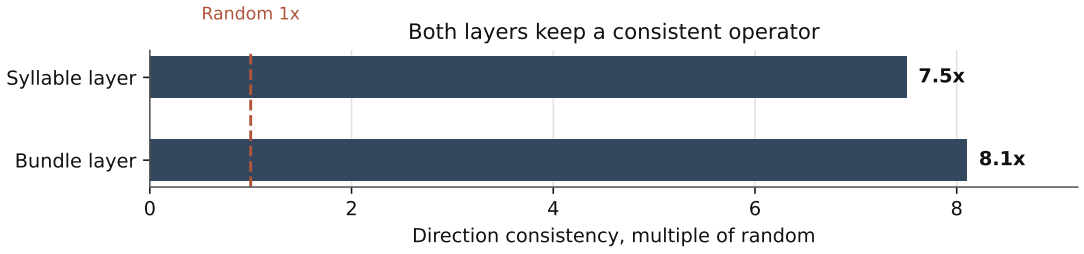
Measured

As measured, the operation plane of bundle-layer tokens also acts as consistent operators.

- Direction consistency : the direction consistency of bundle tokens is 0.252, 8.1 times the random baseline of 0.031 (syllable tokens: 7.5 times). The operator emergence seen in Part 3 [3] is preserved at the bundle layer.
- Operation orthogonality : 77 percent of pairs of distinct operations among bundle tokens are nearly orthogonal (syllables: 79 percent), and same-direction pairs are 0 percent. Each bundle preserves its own operation.

That is, enlarging the vocabulary with bundles does not break the data-operation orthogonal structure.

Figure 3-2. Direction consistency of the syllable layer and the bundle layer



Both the syllable layer (7.5 times) and the bundle layer (8.1 times) show direction consistency far above random. The data-operation orthogonal structure is preserved even when the vocabulary is enlarged with bundles.

Reproduce

The figures in Sections 3.1 and 3.2 are reproduced with the following one-liner after completing the preparation in Appendix C.

```
cd probe && python3 paper5_cache_token_probe.py
```

In the output, vocabulary 10724 (2724 syllables + 8000 bundles), the 7.3-fold saving of the low-rank head, bundle-layer direction consistency of 8.1 times, and orthogonal-pair rates of 79% and 77% correspond to the figures in the text.

3.3 Do the savings travel with quality, and the bundles-off diagnostic

The practical benefit of the bundle dictionary is encoding the same text with fewer tokens. To check whether that saving travels together with prediction quality, we passed two sets of holdout text unused in training (30 fragments each, 6,571 characters in total) through the same frozen model under two encodings. Bundles on is the cache dictionary's greedy longest-match encoding; bundles off is the encoding unrolled into syllable atoms only. Because the token units differ, per-token cross-entropy is not comparable, so we normalized to bits per character by dividing by the number of characters covered by the predicted tokens. Every fragment fits within a single context window under both encodings, so there was no context truncation on either side.

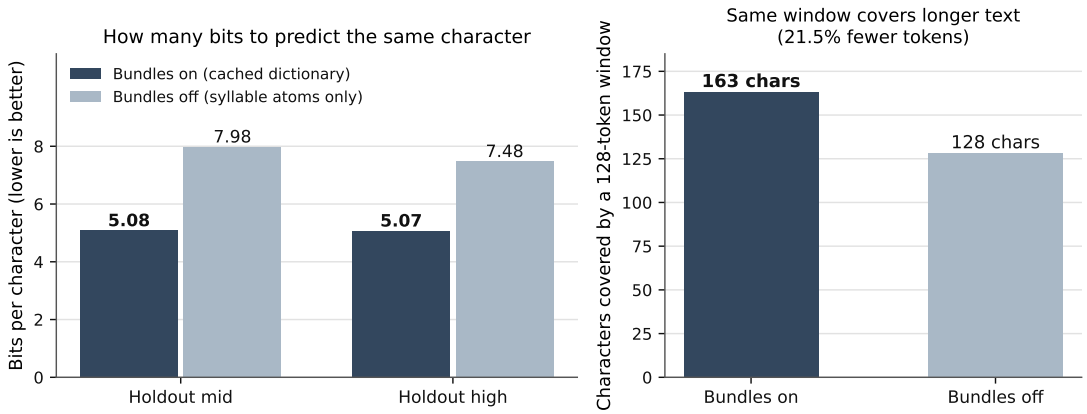
Measured

The normal path, bundles on, runs at 5.08 bits per character (middle-school holdout) and 5.07 bits per character (high-school holdout) while encoding the same text with 21.5% fewer tokens. Converted to the number of characters covered by a 128-token window, on covers about 163 characters vs. 128 for off; this is an arithmetic conversion of the token-to-character ratio, and in this measurement no context-window difference affected the bit figures.

Evaluation set	Bundles on	Bundles off	Off (atom-vocabulary renormalization)	Per-fragment difference
Middle-school holdout (30 fragments)	5.08	7.98	7.58	+2.90±0.95, positive 30/30
High-school holdout (30 fragments)	5.07	7.48	7.07	+2.41±0.82, positive 30/30

The bundles-off figures should be read as an out-of-distribution diagnostic, not a quality comparison. This model was trained on the bundles-on stream, and because greedy longest match is deterministic, forms in which frequent words are spelled out in atoms do not exist in training by construction. Indeed, the off-mode collapse concentrates at word-internal positions: word-internal positions run at 11.49 bits per token while boundary positions run at 6.69. With bundles off (the atom stream), the model assigns an average of 17.6% of its probability mass to bundle tokens, and at 12.9% of positions the max-probability token is a bundle. That is, the model is trying to emit bundles at atom positions, which means the knowledge of these words resides in the bundle path.

Figure 3-3. Bits per character with bundles on and off, and characters covered per window



Left: bits per character on the same holdout text; the normal path, bundles on, runs at 5.08 and 5.07 bits. Bundles off is an out-of-distribution diagnostic, and renormalizing to the atom vocabulary lowers it to 7.58 and 7.07. Right: characters per window is an arithmetic conversion of the token-to-character ratio.

We state the interpretive limits of this measurement. Because there is no control model trained with bundles off, we make no causal claim that bundles are the cause of quality. A model trained atom-only could do far better than the off figures. What this measurement shows is only that the normal path operates together with the savings, and that this system's word knowledge resides one-directionally in the bundle path. Bits per character is the code length of the single greedy longest-match segmentation and hence an upper bound on the true probability summed over segmentations, and the saving rate and bit figures are relative to these two holdout sets.

Reproduce

The quality contrast of this subsection is reproduced with the following one-liner after completing the preparation in Appendix C.

```
cd probe && python3 paper5_quality_contrast_probe.py
```

In the output, bundles-on 5.08 and 5.07 bits, the off diagnostic 7.98 and 7.48 (atom-

vocabulary renormalization 7.58 and 7.07), word-internal 11.49 vs. boundary 6.69 bits, and the 21.5% token saving correspond to the figures in the text.

4. Limitations and Next Measurement Tasks

The baseline is a reference class of one-person development, a self-built exact-adjoint back-propagation engine, and a single consumer GPU. The items below are not defects but next measurement tasks.

- Comparison against standard tokenization : placing the cache dictionary side by side with standard subword tokenization at the same vocabulary size, to confirm that it does at least as well, is a next task.
- Dynamic growth : dynamic growth of the bundle dictionary during training is designed; its measurement is a next task. Dynamic embedding, in which the embedding and the head grow as the vocabulary grows, is what this dynamic-growth design covers.
- Compression and downstream tasks : Section 3.3 measured savings and quality traveling together in bits per character, but side-by-side comparison on downstream tasks such as question answering, and against a control model trained atom-only, is a next task.

5. Conclusion

This paper presented a cache-dictionary tokenization that stacks a bundle dictionary on syllable atoms, and a low-rank output head that makes vocabulary expansion nearly free. At vocabulary 10724, the low-rank head uses about 7.3 times fewer parameters than the dense head, and bundle-layer tokens preserve the orthogonal structure of the data plane and the operation plane with a direction consistency of 8.1 times. The vocabulary grows without breaking the structure established by the earlier papers. The savings travel with quality: the normal path predicts the holdout at 5.07 to 5.08 bits per character while saving 21.5% of tokens, and bundles off collapses word-internally, showing that word knowledge resides in the bundle path. A causal comparison against a control model trained atom-only remains as a next task. Why the orthogonality must hold as the vocabulary grows is a reason that lies outside this paper's efficiency arithmetic; the end of the series states it.

6. Series Preview

Series preview

- Part 6 — Control signals emerging from self-distribution and geometric monitoring : treats, as measured statistics, the signals by which the model reads its own output

distribution and embedding geometry to route certain, vague, and unknown states.

- Final part (unfinished) — Combining similarity consolidation with inference traversal : self-utterance weaving together data-plane similarity consolidation (Part 2) and operation-plane inference traversal. Not yet covered, as the final assembly is incomplete.

References

[1] Han, H. (2026). Banya Framework: An Axiom-Based Science Mining Engine for Deriving Fundamental Physical Constants. Zenodo. https://doi.org/10.5281/zenodo.20301304 . (Background natural philosophy of this series; the prior axiom document)

[2] Han, H. (2026). A Language-Model Training Engine Using Hand-Derived Closed-Form Exact Adjoints Without Automatic Differentiation. Banya Research Series, Part 1. Zenodo. https://doi.org/10.5281/zenodo.21385447

[3] Han, H. (2026). Orthogonal Data-Operation Tokens and Convolutional Scale-Depth Structure: Label-Free Emergence of Operation-as-Weight. Banya Research Series, Part 3. Zenodo. https://doi.org/10.5281/zenodo.21385676

Appendix A. Vocabulary Composition

Table A-1. The measured model's two-layer cache dictionary

Layer	Count	Content
Syllable atoms	2724	Korean phonetic units
Bundle dictionary	8000	Frequent words as whole units
Total vocabulary	10724	Shared by input and output

Appendix B. Measurement Environment

Table B-1. Computer specifications used for the measurements

Item	Specification
Graphics processing unit (GPU)	NVIDIA GeForce RTX 3090 Ti (24 GB)
Operating system	Ubuntu 24.04.4 LTS
Software	Python 3.12.3, numpy 2.5.1, scipy 1.18.0, cupy 14.1.1, CUDA 13.3.1
Measurement target	Vocabulary and the syllable and bundle operation planes of the frozen elementary-stage model

The measured figures in Section 3 were taken in the environment above. The low-rank head parameter count is a structural computation.

Appendix C. Reproduction Guide

The measured figures in this paper are reproduced with the public reproduction package. Follow the five steps below in order.

1. Requirements. An NVIDIA graphics processing unit (GPU) with CUDA and Python 3.12 are required. Python packages are installed from the unzipped folder with the following command. For cupy, use the distribution matching the installed CUDA version (e.g., cupy-cuda13x for CUDA 13).

```
pip install -r requirements.txt
```

2. Get the code. Download the zip archive and unpack it. To get it as a repository, use the github.com/ubmscoin/banya/banya_ai_paper_code folder.

```
wget https:
//ubmscoin.github.io/banya/banya_ai_paper_code/banya_ai_paper_code.zip
unzip banya_ai_paper_code.zip && cd banya_ai_paper_code
```

3. Get the models. The three frozen models (471 MB in total) are on Zenodo at doi.org/

10.5281/zenodo.21383724. The following one-liner performs the download together with integrity verification (sha256 file-fingerprint check). To do it by hand, download the three files from the record above, place them in the model folder, and verify with `sha256sum -c checksums.sha256`.

```
cd model && bash download.sh && cd ..
```

4. Build the corpora. The corpora are produced by generators, with no raw-text distribution. The following one-liner generates all 23 corpora into the banya_world_data folder.

```
cd corpus_build && bash build_all.sh && cd ..
```

5. Reproduce the measurements. The probes in the probe folder reproduce the figures of each section. The figures in this paper are covered by the probes below. Target values are printed alongside the output, so they can be checked directly against the text.

Probe	Figures reproduced	Output to check
paper5_cache_token_probe.py	3.1 cost, 3.2 bundle-layer structure	vocabulary 10724, low-rank 7.3-fold, bundle-layer consistency 8.1 times, orthogonal pairs 79% and 77%
paper5_quality_contrast_probe.py	3.3 savings and quality	on 5.08 and 5.07 bits, off 7.98 and 7.48, renormalized 7.58 and 7.07, saving 21.5%

The package's folder layout and per-file descriptions are on the table-of-contents page.

Banya Research Series, Part 5 (cache-dictionary tokenization, dynamic embedding). All figures in this paper were measured directly under the Appendix B environment or computed from the structure. The background natural philosophy is cited as the prior axiom document [1] but is not the basis of this paper's claims. Parts 1 and 3 are cited. Emergent control signals are treated in Part 6, and the self-utterance combination in the final part.