

The tables below provide side-by-side comparisons between corresponding claims of the '775 patent.

Differences for the compared claim language on the right side of each chart are indicated with ~~cross-through~~ and underline.

I. '775 PATENT | INDEPENDENT CLAIMS 1 AND 10

#	775 PAT CLAIM 1	#	775 PAT CLAIM 10
775_1[PRE]	1. A method of dynamically executing batches of requests on one or more execution engines running a machine-learning transformer model, comprising:	775_10[PRE]	A ~~method of~~ <u>non-transitory computer-readable storage medium storing computer program instructions executable to perform operations</u> for dynamically executing batches of requests on one or more execution engines running a machine-learning transformer model, <u>the operations</u> comprising:
775_1[A]	receiving, by a serving system, one or more requests for execution, the serving system including a scheduler and one or more execution engines each coupled to access a machine-learning transformer model including at least a set of decoders;	775_10[A]	receiving, by a serving system, one or more requests for execution, the serving system including a scheduler and one or more execution engines each coupled to access a machine-learning transformer model including at least a set of decoders;
775_1[B]	scheduling, by the scheduler, a batch of requests including the one or more requests for execution on an execution engine;	775_10[B]	scheduling, by the scheduler, a batch of requests including the one or more requests for execution on an execution engine;

#	775 PAT CLAIM 1	#	775 PAT CLAIM 10
775_1[C]	generating, by the execution engine, a first set of output tokens by applying the transformer model to a first set of inputs for the batch of requests, wherein applying the transformer model comprises applying at least one batch operation to one or more input tensors associated with the batch of requests;	775_10[C]	generating, by the execution engine, a first set of output tokens by applying the transformer model to a first set of inputs for the batch of requests, wherein applying the transformer model comprises applying at least one batch operation to one or more input tensors associated with the batch of requests;
775_1[D]	receiving, by a request processor, a new request from a client device, the new request including a sequence of input tokens;	775_10[D]	receiving, by a request processor, a new request from a client device, the new request including a sequence of input tokens;
775_1[E]	scheduling, by the scheduler, a second batch of requests additionally including the new request for execution on the execution engine, the second batch of requests scheduled responsive to determining that the execution engine has memory available to execute the second batch of requests, wherein in a second set of inputs for the second batch of requests, a length of the sequence of input tokens for the new request is different from a length of an input for	775_10[E]	scheduling, by the scheduler, a second batch of requests additionally including the new request for execution on the execution engine, the second batch of requests scheduled responsive to determining that the execution engine has memory available to execute the second batch of requests, wherein in a second set of inputs for the second batch of requests, a length of the sequence of input tokens for the new request is different from a length of an input for

#	775 PAT CLAIM 1	#	775 PAT CLAIM 10
	at least one request other than the new request; and		at least one request other than the new request; and
775_1[F]	generating, by the execution engine, a second set of output tokens by applying the transformer model to the second set of inputs for the second batch.	775_10[F]	generating, by the execution engine, a second set of output tokens by applying the transformer model to the second set of inputs for the second batch.

II. '775 PATENT | DEPENDENT CLAIMS 2 AND 11

#	775 PAT CLAIM 2	#	775 PAT CLAIM 11
775_2[A]	2. The method of claim 1, further comprising: responsive to determining that a request in the first batch of requests has been completed, providing output tokens generated for the completed request to a client device as a response to the request, and	775_11[A]	11. The ~~method of claim 1~~ , <u>non-transitory computer-readable storage medium of claim 10, the operations</u> further comprising: responsive to determining that a request in the first batch of requests has been completed, providing output tokens generated for the completed request to a client device as a response to the request, and
775_2[B]	wherein the second batch of requests includes at least one of the remaining requests from the one or more requests and the new request.	775_11[B]	wherein the second batch of requests includes at least one of the remaining requests from the one or more requests and the new request.

III. '775 PATENT | DEPENDENT CLAIMS 3 AND 12

#	775 PAT CLAIM 3	#	775 PAT CLAIM 12
775_3	3. The method of claim 2, wherein the request is associated with a cache memory in the execution engine dedicated for storing an internal state for the request, and responsive to determining that the request has been completed, freeing the dedicated cache memory for the request in the execution engine.	775_12	12. The ~~method of claim 1~~ , <u>non-transitory computer-readable storage medium of claim 10</u> , wherein the request is associated with a cache memory in the execution engine dedicated for storing an internal state for the request, and responsive to determining that the request has been completed, freeing the dedicated cache memory for the request in the execution engine.

IV. '775 PATENT | DEPENDENT CLAIMS 4 AND 13

#	775 PAT CLAIM 4	#	775 PAT CLAIM 13
775_4	4. The method of claim 1, wherein the input for the at least one request is an output token from the first set of output tokens for the at least one request, and wherein a length of the sequence of input tokens for the new request is different from a length of the output token for the at least one request.	775_13	13. The ~~method~~ <u>non-transitory computer-readable storage medium</u> of claim 10, wherein the input for the at least one request is ~~an~~ <u>at least one</u> output token from the first set of output tokens for the at least one request, and wherein a length of the sequence of input tokens for the new request is different from a length of the <u>at least one</u> output token for the at least one request.

V. '775 PATENT | DEPENDENT CLAIMS 5 AND 14

#	775 PAT CLAIM 5	#	775 PAT CLAIM 14
775_5[A]	5. The method of claim 4, wherein the execution engine includes a cache memory for maintaining a key cache tensor for storing keys and a value cache tensor for storing values for the at least one request, and	775_14[A]	14. The ~~method of claim 4~~ , <u>non-transitory computer-readable storage medium of claim 13</u> , wherein the execution engine includes a cache memory for maintaining a key cache tensor for storing keys and a value cache tensor for storing values for the at least one request, and
775_5[B]	wherein after scheduling the second batch of requests, allocating, by the execution engine, a new cache memory dedicated to maintaining a key cache tensor and a value cache tensor for the new request.	775_14[B]	wherein after scheduling the second batch of requests, allocating, by the execution engine, a new cache memory dedicated to maintaining a key cache tensor and a value cache tensor for the new request.

VI. '775 PATENT | DEPENDENT CLAIMS 6 AND 15

#	775 PAT CLAIM 6	#	775 PAT CLAIM 15
775_6	6. The method of claim 5, wherein after generating the second set of output tokens, a length of the key cache tensor for the at least one request is different from a length of a key cache tensor for the new request, and a length of the value cache tensor for the at least one request is different from a length of a value cache tensor for the new request.	775_15	15. The ~~method of claim 4~~ , <u>non-transitory computer-readable storage medium of claim 14</u> , wherein after generating the second set of output tokens, a length of the key cache tensor for the at least one request is different from a length of a key cache tensor for the new request, and a length of the value cache tensor for the at least one request is different from a length of a value cache tensor for the new request.

VII. '775 PATENT | DEPENDENT CLAIMS 7 AND 16

#	775 PAT CLAIM 7	#	775 PAT CLAIM 16
775_7	7. The method of claim 1, after receiving the new request from the client device, determining, by the scheduler, that there is insufficient memory available to execute the second batch of requests on a second execution engine different from the execution engine, and responsive to the determination for the second execution engine, determining	775_16	16. The ~~method of claim 1~~ , <u>non-transitory computer-readable storage medium of claim 10</u> , after receiving the new request from the client device, determining, by the scheduler, that there is insufficient memory available to execute the second batch of requests on a second execution engine different from the execution engine, and responsive to the

#	775 PAT CLAIM 7	#	775 PAT CLAIM 16
	whether the execution engine has the memory available to execute the second batch of requests.		determination for the second execution engine, determining whether the execution engine has the memory available to execute the second batch of requests.

VIII. '775 PATENT | DEPENDENT CLAIMS 8 AND 17

#	775 PAT CLAIM 8	#	775 PAT CLAIM 17
775_8	8. The method of claim 1, wherein the execution engine is configured as a graphics processing unit (GPU) or a tensor processing unit (TPU).	775_17	17. The ~~method of claim 1~~ , <u>non-transitory computer-readable storage medium of claim 10</u> , wherein the execution engine is configured as a graphics processing unit (GPU) or a tensor processing unit (TPU).

IX. '775 PATENT | DEPENDENT CLAIMS 9 AND 18

#	775 PAT CLAIM 9	#	775 PAT CLAIM 18
775_9	9. The method of claim 1, wherein each token in the sequence of input tokens represents a text unit.	775_18	18. The ~~method of claim 1~~ , <u>non-transitory computer-readable storage medium of claim 10</u> , wherein each token in the sequence of input tokens represents a text unit.