The tables below provide side-by-side comparisons between corresponding claims of the ’

Differences for the compared claim language on the right side of each chart are indicated with ~~cr~~

underline.

## I. ’775 PATENT | INDEPENDENT CLAIMS 1 AND 10

| # | 775 PAT \| CLAIM 1 | # | 775 PAT \| C |
|---|---|---|---|
| **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~~non-tra readable storage me computer program i executable to perfor for dynamically exe requests on one or m engines running a m transformer model, operations comprisi |
| **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 servi more requests for ex serving system inclu and one or more exe each coupled to acc learning transforme at least a set of deco |
| **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 s of requests includin requests for executio execution engine; |

| # | 775 PAT \| CLAIM 1 | # | 775 PAT \| C |
|---|---|---|---|
| **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 e first set of output to the transformer mod inputs for the batch wherein applying th model comprises ap batch operation to o tensors associated w 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 requ new request from a new request includi 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 s second batch of requ including the new re execution on the exe the second batch of scheduled responsiv that the execution en available to execute of requests, wherein inputs for the secon requests, a length of input tokens for the different from a leng |

| # | 775 PAT \| CLAIM 1 | # | 775 PAT \| C |
|---|---|---|---|
| | at least one request other than the new request; and | | at least one request 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 e second set of output applying the transfo second set of inputs batch. |

## II. ’775 PATENT | DEPENDENT CLAIMS 2 AND 11

| # | 775 PAT \| CLAIM 2 | # | 775 PAT \| C |
|---|---|---|---|
| 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 c~~ transitory computer- medium of claim 10 operations further co responsive to determ request in the first b has been completed tokens generated for request to a client de response to the requ |
| 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 includes at least one requests from the on requests and the new |

## III. ’775 PATENT | DEPENDENT CLAIMS 3 AND 12

| # | 775 PAT \| CLAIM 3 | # | 775 PAT \| C |
|---|---|---|---|
| **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 c~~ transitory computer- medium of claim 10 request is associated memory in the exec dedicated for storing for the request, and determining that the completed, freeing t cache memory for th execution engine. |

## IV. ’775 PATENT | DEPENDENT CLAIMS 4 AND 13

| # | 775 PAT \| CLAIM 4 | # | 775 PAT \| C |
|---|---|---|---|
| **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~~ non computer-readable s claim 10, wherein th least one request is ~~a~~ one output token fro output tokens for the request, and wherein sequence of input to request is different f the at least one outp least one request. |

## V. ’775 PATENT | DEPENDENT CLAIMS 5 AND 14

| # | 775 PAT \| CLAIM 5 | # | 775 PAT \| C |
|---|---|---|---|
| **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 c~~ transitory computer- medium of claim 13 execution engine inc memory for maintai tensor for storing ke cache tensor for stor at least one request, |
| **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 schedu batch of requests, al execution engine, a memory dedicated t key cache tensor an tensor for the new re |

# Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

## Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

## Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

## Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

## API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

**LAW FIRMS**

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

**FINANCIAL INSTITUTIONS**

Litigation and bankruptcy checks for companies and debtors.

**E-DISCOVERY AND LEGAL VENDORS**

Sync your system to PACER to automate legal marketing.

**WHAT WILL YOU BUILD?** | **sales@docketalarm.com** | **1-866-77-FASTCASE**