Variant Effect Predictor Web interface

Use the VEP online to analyse your variants through a simple point-and-click interface.

The web interface allows you to access the key features of the VEP without using the command line. Interactively filter your results to find the data you want. Download your results in multiple data formats, easily share your results with others, and integrate your variation data with the powerful Ensembl web browser.

If you use the VEP in your work, please cite **McLaren et. al.** (doi:10.1093/bioinformatics/btq330)

Any questions? Send an email to the Ensembl developer's mailing list, dev@ensembl.org or contact the Ensembl Helpdesk at helpdesk@ensembl.org.

Documentation contents

Download documentation in PDF format

Variant Effect Predictor ✎ Input form

When you reach the VEP web interface, you will be presented with a form to enter your data and alter various options.

Data input

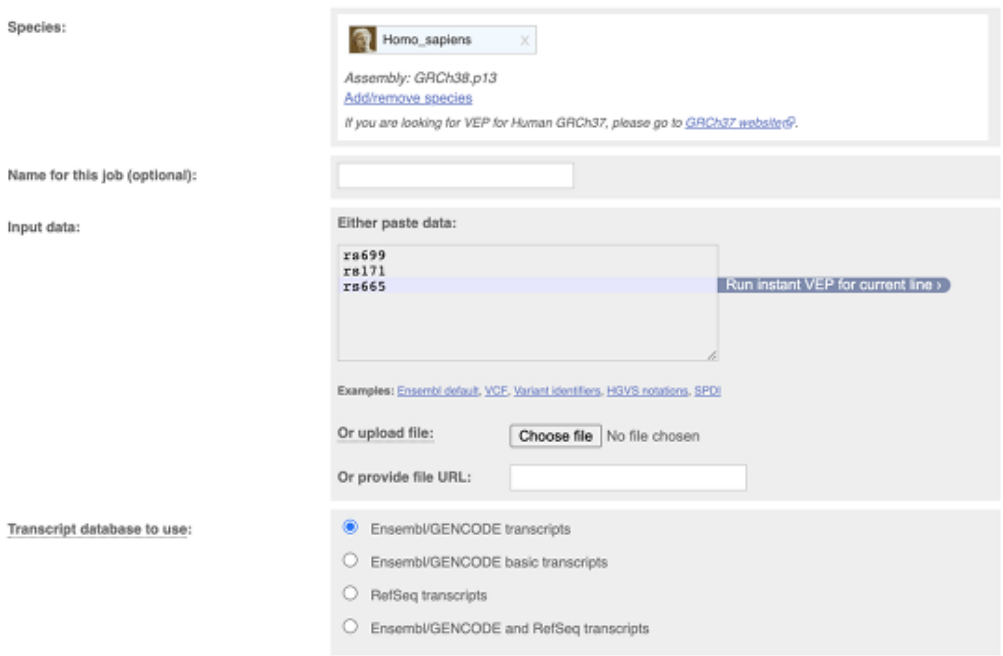
1. First select the correct species for your data. Ensembl hosts many vertebrate genomes; genomes for plants, protists and fungi can be found at Ensembl Genomes.
2. You can optionally choose a name for the data you upload - this can make it easier for you to identify jobs and files that you have uploaded to the VEP at a later point.
3. You have three options for uploading your data:
   - **File upload** - click the "Choose file" button and locate the file on your system
   - **Paste file** - simply copy and paste the contents of your file into the large text box
   - **File URL** - point the VEP to a file hosted on a publically accessible address. This can be either a **http://** or **ftp://** address.

   Once you have uploaded some data, you can select it as the input for future jobs by choosing the data from the drop down menu.

   The format of your data is automatically detected; see the examples or the input format documentation.
4. For pasted data you can get an instant preview of the results of your first variant by clicking the button that appears when you paste your data. This quickly shows you the consequence type, the IDs of any overlapping variants, genes, transcripts and regulatory features, as well as SIFT and PolyPhen predictions. To see the full results set submit your job as normal.
5. For some species you can select which transcript database to use. The default is to use Ensembl transcripts, which offer the most rich annotation through VEP.

   GENCODE Basic is a subset of the GENCODE gene set, and is intended to provide a simplified, high-quality subset of the GENCODE transcript annotations that will be useful to the majority of users. GENCODE Basic includes all genes in the GENCODE gene set, with a representative subset of the transcripts (splice variants).

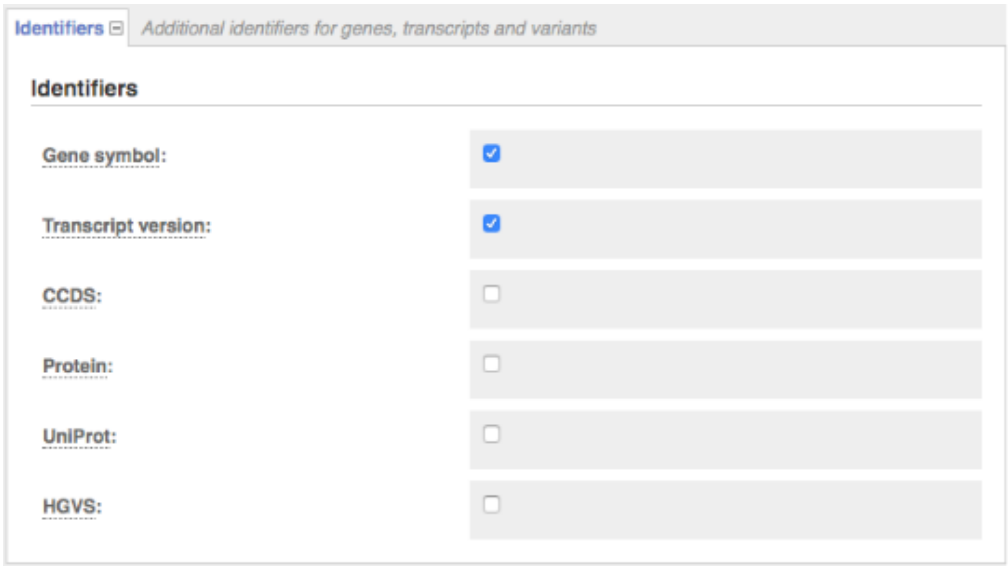
   You can also select to use RefSeq transcripts from the otherfeatures database; note though that these transcripts are simply aligned to the reference genome and the database is missing much of the annotation found when using the main Ensembl database (e.g. protein domains, CCDS identifiers).



Identifiers

VEP can provide additional identifiers for genes, transcripts, proteins and variants.

- **Gene symbol**
  Add the gene symbol for the gene to the output. This will typically be, for example, the HGNC identifier for genes in human. Equivalent to --symbol in the VEP script.
- **Transcript version**
  Add the transcript version to the transcript identifier. Equivalent to --transcript_version.
- **CCDS**
  Add the Consensus CDS transcript identifier where available. Equivalent to --ccds.
- **Protein**
  Add the Ensembl protein identifer (ENSP). Equivalent to --protein.
- **UniProt**
  Add identifiers for translated protein products from three UniProt-related databases (SWISSPROT, TREMBL and UniParc). Equivalent to --uniprot.
- **HGVS**
  Generate HGVS identifiers for your input variants relative to the transcript coding sequence (HGVSc) and the protein sequence (HGVSp). Equivalent to --hgvs.



Variants and frequency data

VEP can also search the Ensembl database for known variants that are co-located with variants from your input data.

- **Find co-located known variants** - report known variants from the Ensembl Variation database that overlap with your input. A list of variant sources imported can be viewed here. Note that this feature is only available for species with an Ensembl Variation database. Equivalent to --check_existing.

  VEP will by default compares the alleles of your input variant to that of the existing variant; VEP will only report the existing variant ID if none of the alleles in your input variant are novel.

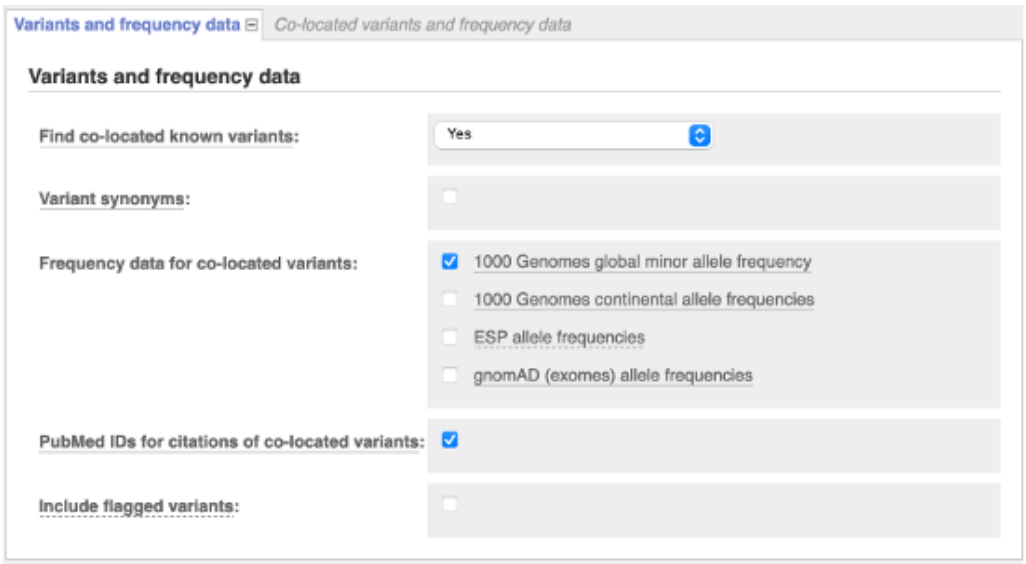
  For example, if your input variant has alleles A/G, and the existing variant has alleles A/T, then the existing variant will not be reported. If instead your input variant has alleles A/T, then the existing variant will be reported.

  To disable this allele matching, select the option "Yes but don't compare alleles" for the option "Find co-located known variants".

  For known variants VEP can also provide PubMed IDs of publications citing the variant (equivalent to --pubmed).
- **Variant synonyms**
  Report known synonyms for co-located variants.
- **Frequency data for co-located variants**
  VEP can also report allele frequency (AF) data for existing variants from several major genotyping projects, the 1000 Genomes Project, the NHLBI-ESP and gnomAD; this only applies when you have selected human as your species.
  - **1000 Genomes global** - the combined phase 3 population (i.e. all individuals from all populations). Equivalent to --af

- **1000 Genomes continental** - the four continent-level populations - AFR (African), AMR (American), ASN (Asian) and EUR (European). Equivalent to --af_1kg
- **ESP** - AA (African American) and EA (European American) populations. Equivalent to --af_esp
- **gnomAD** - combined, AFR, AMR, ASJ, EAS, FIN, NFE, OTH, SAS populations. Equivalent to --af_gnomad

- **PubMed IDs for citations of co-located variants**
  Report the PubMed IDs of any publications that cite the co-located variant(s).
- **Include flagged variants**
  Variants flagged as failed by the Ensembl Variation quality control.



Additional annotations

- **Transcript biotype**
  Add the transcript biotype to the output. Equivalent to --biotype in the VEP script.
- **Exon and intron numbers**
  Report the exon or intron number that a variant falls in as NUMBER / TOTAL, i.e. exon 2/5 means the variant falls in the 2nd of 5 exons in the transcript. Equivalent to --numbers.
- **Transcript support level**
  Report the transcript support level of the overlapped transcript. Equivalent to --tsl.
- **APPRIS**
  Report the APPRIS score of the overlapped transcript. Equivalent to --appris.
- **Identify canonical transcripts**
  Add a flag to the output indicating if the reported transcript is the canonical transcript for the gene. Equivalent to --canonical.
- **Upstream/Downstream distance (bp)**
  Change the distance to assign the upstream and downstream consequences. Equivalent to --distance.
- **miRNA structure**
  Determines where in the secondary structure of a miRNA a variant falls. Equivalent to the VEP plugin miRNA.

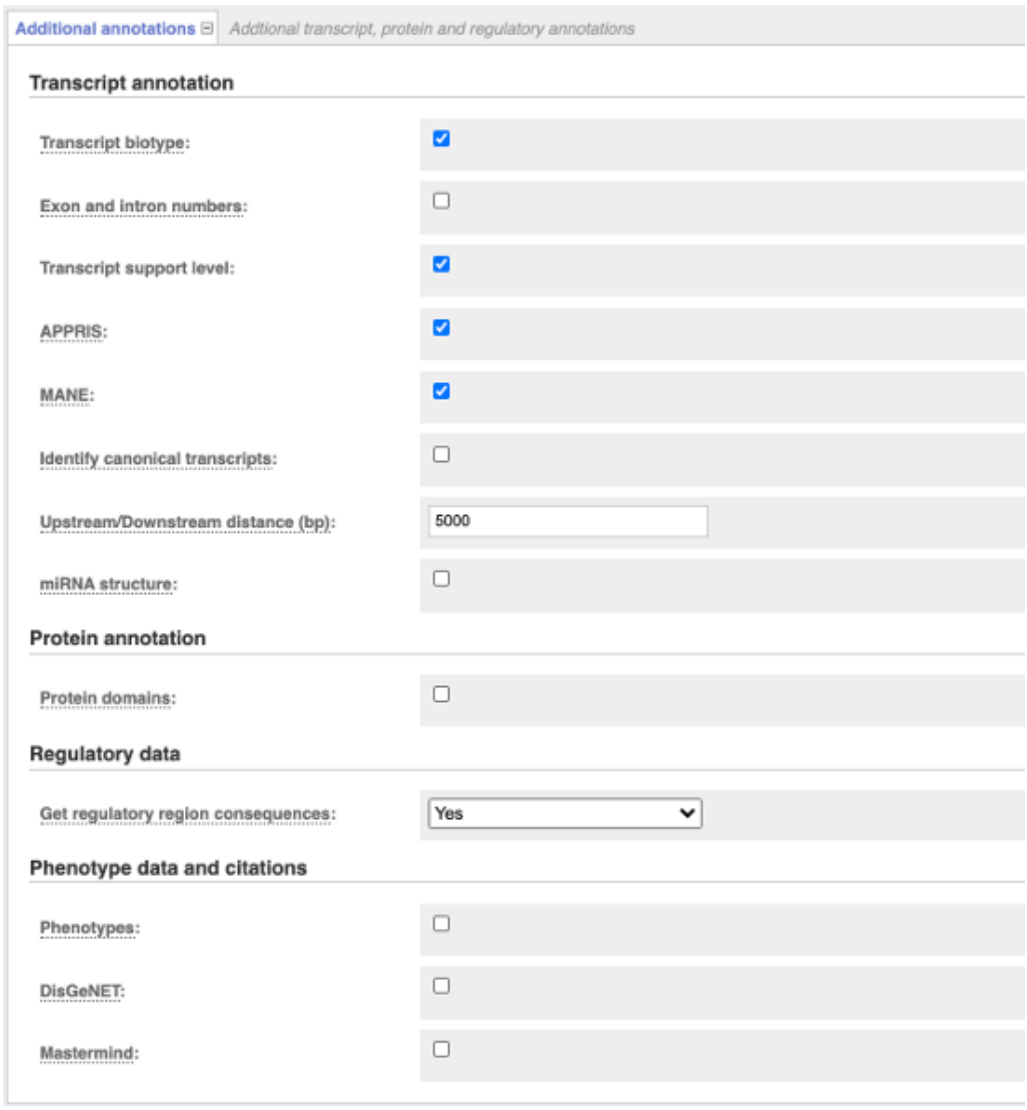
- **Protein domains**
  Report protein domains from PDBe, Pfam, Prosite and InterPro that overlap input variants. Equivalent to --domains.

- **Get regulatory region consequences**
  In addition to predicting consequences with overlapping transcripts, VEP can find overlaps with known regulatory regions as determined in the Ensembl Regulatory build.
  Using this option, VEP will also report if a variant falls in a transcription factor binding motif, and give a score that reflects whether the altered motif sequence is more or less similar to the consensus.
  Get regulatory consequences is equivalent to --regulatory.

- **Phenotypes**
  Report the phenotypic data overlapping the genomic features. This functionality is provided by the Phenotypes plugin. For more information on the imported phenotypic data for genes, variation and QTLs see our phenotype documentation.
  Note: This web functionality is not reporting cancer phenotypic data this release. However the cancer phenotypic data is available in the command line version.

- **DisGeNET**
  Report Variant-Disease-PMID associations from the DisGeNET database. This functionality is provided by the DisGeNET plugin.
  Note: This web functionality is reporting the unique disease names.

- **Mastermind**
  Uses the Mastermind Genomic Search Engine to report variants that have clinical evidence cited in the medical literature. This functionality is provided by the Mastermind plugin.
  Note: This web functionality is only reporting the URL to the Mastermind Genomic Search Engine webpage.



Predictions

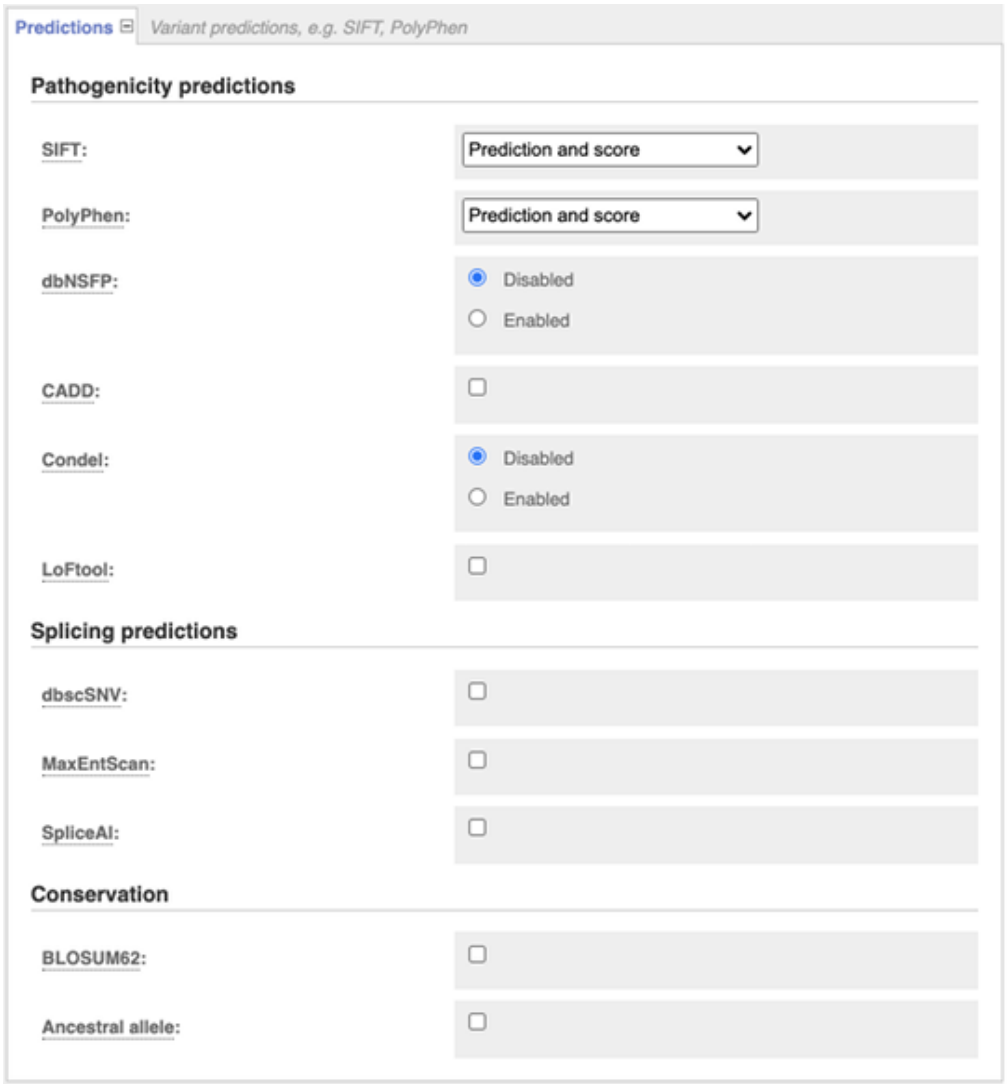
- **SIFT predictions**
  SIFT predicts whether an amino acid substitution affects protein function based on sequence homology and the physical properties of amino acids. Only available in popular species. For both SIFT and PolyPhen VEP can report either a score between 0 and 1, a prediction in words, or both. Equivalent to --sift.

- **PolyPhen predictions**
  PolyPhen is a tool which predicts possible impact of an amino acid substitution on the structure and function of a human protein using straightforward physical and comparative considerations. Equivalent to --polyphen.

- **dbNSFP**
  Retrieves data for missense variants from dbNSFP. Equivalent to the VEP plugin dbNSFP.
- **Condel**
  Calculates the Consensus Deleteriousness (Condel) score for a missense mutation based on the pre-calculated SIFT and PolyPhen-2 scores. Equivalent to the VEP plugin Condel.
- **LoFtool**
  Provides a rank of genic intolerance and consequent susceptibility to disease based on the ratio of Loss-of-function (LoF)to synonymous mutations. Equivalent to the VEP plugin LoFtool.

- **dbscSNV**
  Retrieves data for splicing variants from dbscSNV. Equivalent to the VEP plugin dbscSNV.
- **MaxEntScan**
  Get splice site predictions from MaxEntScan. Equivalent to the VEP plugin MaxEntScan.
- **SpliceAI**
  Pre-calculated annotations from SpliceAI a deep neural network, developed by Illumina, Inc that predicts splice junctions from an arbitrary pre-mRNA transcript sequence. Used for non-commercial purposes. This functionality is provided by the SpliceAI plugin.

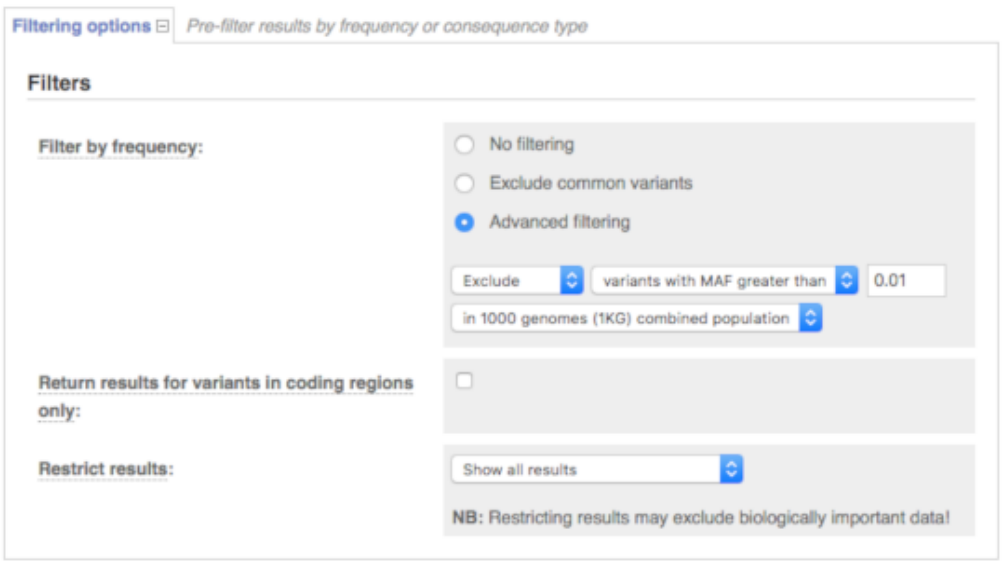
- **BLOSUM62**
  Looks up the BLOSUM 62 substitution matrix score for the reference and alternative amino acids predicted for a missense mutation. Equivalent to the VEP plugin Blosum62.
- **Ancestral allele**
  Retrieves ancestral allele sequences from a FASTA file. Ensembl produces FASTA file dumps of the ancestral sequences of key species. Equivalent to the VEP plugin AncestralAllele.



Filtering options

VEP allows you to pre-filter your results e.g. by MAF or consequence type. Note that it is also possible to perform equivalent operations on the results page for VEP, so if you aren't sure, don't use any of these options!

- **By frequency**
  Filter variants by minor allele frequency (MAF). Two options are provided:
  - **Exclude common variants**
    Filter out variants that are co-located with an existing variant that has a frequency greater than 0.01 (1%) in the 1000 Genomes global population. Equivalent to --filter_common in the VEP script.
  - **Advanced filtering**
    Enabling this option allows you to specify a population and frequency to compare to, as well whether matching variants should be included or excluded from the results.
- **Return results for variants in coding regions only**
  Exclude variants that don't fall in a coding region of a transcript. Equivalent to --coding_only.
- **Restrict results**
  For many variants VEP will report multiple consequence types - typically this is because the variant overlaps more than one transcript. For each of these options VEP uses consequence ranks that are subjectively determined by Ensembl. This table gives all of the consquence types predicted by Ensembl, ordered by rank. Note that enabling one of these options not only loses potentially relevant data, but in some cases may be scientifically misleading. Options:
  - **Show one selected consequence**
    Pick one consequence type across all those predicted for the variant; the output will include transcript- or feature-specific information. Consequences are chosen by the canonical, biotype status and length of the transcript, along with the ranking of the consequence type according to this table. This is the best method to use if you are interested only in one consequence per variant. Equivalent to --pick.
  - **Show one selected consequence per gene**
    Pick one consequence type for each gene using the same criteria as above. Note that if a variant overlaps more than one gene, output for each gene will be reported. Equivalent to --per_gene.
  - **Show only list of consequences per variant**
    Give a comma-separated list of all observed consequence types for each variant. No transcript-specific or gene-specific output will be given. Equivalent to --summary.
  - **Show most severe per variant**
    Only the most severe of all observed consequence types is reported for each variant. No transcript-specific or gene-specific output will be given. Equivalent to --most_severe.

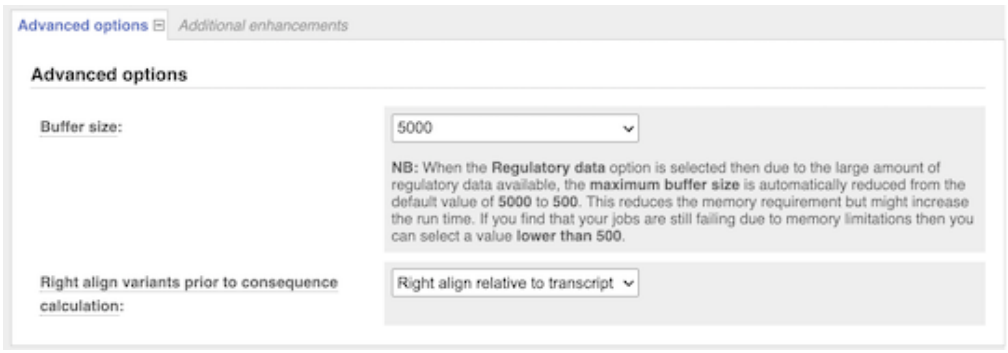


Advanced options

The VEP web interface allows you to use/setup advanced options:

- **Buffer size**
  By default VEP process the variants by blocks of 5000 (i.e. what we call "buffer size").
  In some cases, reducing the size of the blocks (buffer size) could prevent memory issues for large VEP queries (e.g. use of regulatory data, many plugins or custom annotations).
  This is why the maximum buffer size is automatically set to 500 on the VEP Web interface when the "Regulatory data" option is selected.

- **Right align variants prior to consequence calculation**
  By default VEP performs consequence calculation at the given input coordinates.
  Optionally, VEP can shift insertions and deletions found within repeated regions as far as possible in the 3' direction, normalising output.



Jobs

Once you have clicked "Run", your input will be checked and submitted to the VEP as a job. All jobs associated with your session or account are shown in the "Recent Tickets" table. You may submit multiple jobs simultaneously.

The "Jobs" column of the table shows the current status of the job.

- **Queued** - your job is waiting to be submitted to the system
- **Running** - your job is currently running
- **Done** - your job is finished - click the [View results] link to be taken to the results page
- **Failed** - there is a problem with your job - click the magnifying glass icon to see more details

You may delete a job by clicking the trash can icon. If you are logged in to Ensembl, you can save the job by clicking the save icon.

You may also resubmit a job (for example, to re-run with the same data but change some parameters) by clicking the edit icon.

You can see a summary of the options that you selected for your VEP job by clicking on the magnifying glass icon.

Variant Effect Predictor Results

The VEP presents a summary and a detailed results preview on its results page.

Summary

The summary panel on the VEP results page gives a brief overview of the VEP job, along with some basic statistics about the results.

Statistics

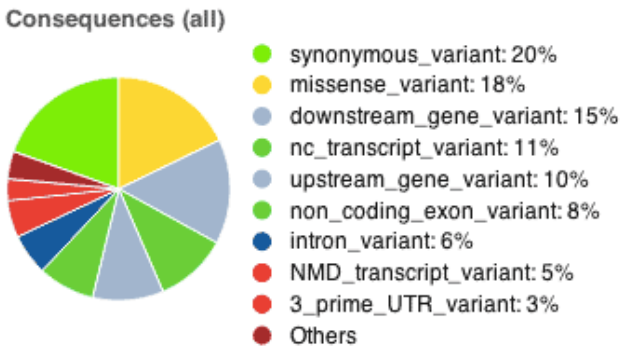
Various statistics are listed in a table, including:

- Variants processed - any variants not parsed by the VEP are not included in this count
- Variants remaining after filtering
- Novel / known variants - the number and percentage of novel variants vs existing variants in the input (see input page documentation)
- Number of overlapped genes, transcripts and regulatory features

Category	Count
Variants processed	498955
Variants remaining after filtering	498955
Novel / known variants	-
Overlapped genes	825
Overlapped transcripts	2888
Overlapped regulatory features	7309

Pie charts

Pie charts are shown detailing the proportion of consequence types called across all variants in the results. The colour scheme of the pie chart matches the colours used to draw variants on the Ensembl region in detail view.



Results preview table

The results table shows one row per transcript and variant. By default all of the columns are shown; to temporarily hide columns, click the blue "Show/hide columns" button and select or deselect the columns you wish to view. The columns you select will be recalled when viewing other jobs.

Hover over a column title to see a description. See the VEP output format documentation for more details on each of the results columns.

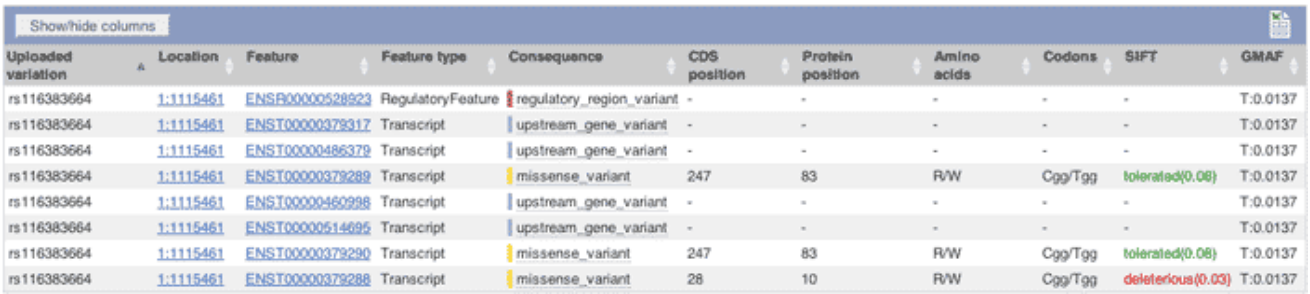
The table can be sorted by any column - click the column header to toggle sorting behaviour.

To download what you see in the table, hover over the spreadsheet icon in the top right corner of the table.

Several columns have special features for the data they contain:

- **Location** - click the link to navigate to the region in detail view for the region surrounding this variant

- **Gene, Feature and Existing Variation** - click the link to bring up a summary view of the gene, transcript, regulatory feature or variation, from which you can navigate to the main Ensembl page for it
- **Consequence** - hover over the consequence name to see the Sequence Ontology definition. See the Ensembl Variation documentation for a full list of consequence types used by the VEP and their definitions
- **SIFT and PolyPhen** - predictions and scores are coloured according to the nature of the prediction, with red indicating deleterious or damaging


Show/hide columns

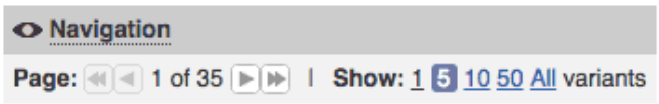
Uploaded variation	Location	Feature	Feature type	Consequence	CDS position	Protein position	Amino acids	Codons	SIFT	GMAF
rs116383664	1:1115461	ENSR00000528923	RegulatoryFeature	regulatory_region_variant	-	-	-	-	-	T:0.0137
rs116383664	1:1115461	ENST00000379317	Transcript	upstream_gene_variant	-	-	-	-	-	T:0.0137
rs116383664	1:1115461	ENST00000486379	Transcript	upstream_gene_variant	-	-	-	-	-	T:0.0137
rs116383664	1:1115461	ENST00000379289	Transcript	missense_variant	247	83	R/W	Cgg/Tgg	tolerated(0.08)	T:0.0137
rs116383664	1:1115461	ENST00000460998	Transcript	upstream_gene_variant	-	-	-	-	-	T:0.0137
rs116383664	1:1115461	ENST00000514695	Transcript	upstream_gene_variant	-	-	-	-	-	T:0.0137
rs116383664	1:1115461	ENST00000379290	Transcript	missense_variant	247	83	R/W	Cgg/Tgg	tolerated(0.08)	T:0.0137
rs116383664	1:1115461	ENST00000379288	Transcript	missense_variant	28	10	R/W	Cgg/Tgg	deleterious(0.03)	T:0.0137

Navigating results

The navigation panel can be used to scroll through pages of results.

By default, the results for five variants are shown. Note that since a variant can overlap multiple transcripts, the table will often show **more than** five rows. To change the number shown, click the appropriate link. Be warned that if your input file is large, it is inadvisable to show all results unless you are sure you have applied sufficient filters - your browser may become unresponsive if it tries to display many thousands of rows in the table.

To navigate between pages of results, use the four arrow icons. Note that when any filters are enabled, it is not possible to navigate to the last page of results as the total number of results cannot be calculated.



Filtering results

You can apply any combination of filters to your results in order to identify interesting data. This is equivalent to using the VEP filtering script on the command line.

To add a filter, simply select the column you wish to filter on, select an "operator", and input a value for the filter to compare to.

To edit a filter, click the pencil icon. To remove a filter, click the cross icon.

When you have added more than one filter, you are given the option to match any or all of the rules shown; click the "Update" button once you have made your selection.

Certain columns when selected have special features:

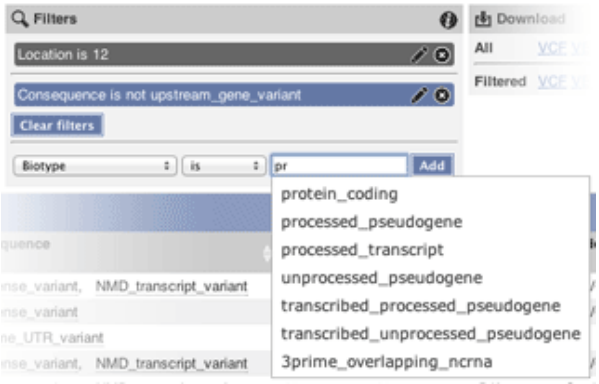
- **Location** - for this column you may enter genomic coordinates in the format "chromosome:start-end". It is also possible to enter just a chromosome, e.g. enter "12" to show only variants on chromosome 12.

  Adding multiple location filters allows you to select multiple regions - location filters are not affected by whether you select "Match all" or "Match any" (see above).

  Users should note that enabling at least one location filter will greatly speed up the return of results (this is because tabix is used behind the scenes).

  Location filters are not affected by the operator selected.
- **Allele, Feature type, Consequence, SIFT, PolyPhen and Biotype** - for these columns, autocomplete will help you fill in the value when you start typing
- **SIFT and PolyPhen** - these columns can contain both text (e.g. a SIFT prediction) and a number (e.g. a frequency value). The VEP allows you to filter on either part of this.

  For example, you may enter "is" and "deleterious" for SIFT to return deleterious predictions, or "<" and "0.1" to find results with a SIFT score less than 0.1.

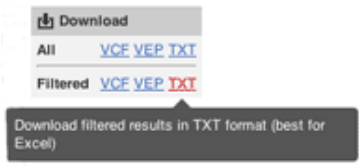


Downloading results

The VEP allows you to download either your full or filtered results set in a choice of data formats.

- **VCF** - VCF is a portable format for variant data. Consequence data is encoded as a series of delimited strings under the "CSQ" key in the VCF INFO field.
- **VEP** - The default VEP output format gives one row per variant and transcript overlap.
- **TXT** - Text format is a tab-delimited format, equivalent to what can be seen in the results table. Note that the columns you select to be visible in the table do not affect the downloaded file - all columns are outputted. This format is best if you intend to import the results into a spreadsheet program such as Microsoft Excel.

You can also send the genes or known variants in your current preview to BioMart. This allows you to easily retrieve any of BioMart's rich data associated with these genes (other database references, GO terms, orthologues/paralogues) and variants (phenotype annotations, synonyms, citations).



Variant Effect Predictor Data formats

Input

Both the web and script version of VEP can use the same input formats. Formats can be auto-detected by the VEP script, but must be manually selected when using the web interface.

VEP can use different input formats:

Default VEP input

The default format is a simple **whitespace-separated** format (columns may be separated by space or tab characters), containing five required columns plus an optional identifier column:

1. **chromosome** - just the name or number, with no 'chr' prefix
2. **start**
3. **end**
4. **allele** - pair of alleles separated by a '/', with the reference allele first
5. **strand** - defined as + (forward) or - (reverse).
6. **identifier** - this identifier will be used in VEP's output. If not provided, VEP will construct an identifier from the given coordinates and alleles.

```
1   881907    881906    -/C   +
5   140532    140532    T/C   +
12  1017956   1017956   T/A   +
2   946507    946507    G/C   +
14  19584687  19584687  C/T   -
19  66520     66520     G/A   +    var1
8   150029    150029    A/T   +    var2
```

An insertion (of any size) is indicated by start coordinate = end coordinate + 1. For example, an insertion of 'C' between nucleotides 12600 and 12601 on the forward strand of chromosome 8 is indicated as follows:

```
8   12601     12600     -/C   +
```

A deletion is indicated by the exact nucleotide coordinates. For example, a three base pair deletion of nucleotides 12600, 12601, and 12602 of the reverse strand of chromosome 8 will be:

```
8   12600     12602     CGT/- -
```

VCF

VEP also supports using VCF (Variant Call Format) version 4.0. This is a common format used by the 1000 genomes project, and can be produced as an output format by many variant calling tools.

Users using VCF should note a peculiarity in the difference between how Ensembl and VCF describe unbalanced variants. For any unbalanced variant (i.e. insertion, deletion or unbalanced substitution), the VCF specification requires that the base immediately before the variant should be included in both the reference and variant alleles. This also affects the reported position i.e. the reported position will be one base before the actual site of the variant.

In order to parse this correctly, VEP needs to convert such variants into Ensembl-type coordinates, and it does this by removing the additional base and adjusting the coordinates accordingly. This means that if an identifier is not supplied for a variant (in the 3rd column of the VCF), then the identifier constructed and the position reported in VEP's output file will differ from the input.

This problem can be overcome with the following:

1. ensuring each variant has a unique identifier specified in the 3rd column of the VCF
2. using VCF format as output (--vcf) - this preserves the formatting of your input coordinates and alleles
3. using --minimal and --allele_number (see Complex VCF entries).

The following examples illustrate how VCF describes a variant and how it is handled internally by VEP. Consider the following aligned sequences (for the purposes of discussion on chromosome 20):

```
Ref: a t C g a // C is the reference base
1 : a t G g a // C base is a G in individual 1
2 : a t - g a // C base is deleted w.r.t. the reference in individual 2
3 : a t CAg a // A base is inserted w.r.t. the reference sequence in individual 3
```

Individual 1

The first individual shows a simple balanced substitution of G for C at base 3. This is described in a compatible manner in VCF and Ensembl styles. Firstly, in VCF:

```
20   3   .   C   G   .   PASS   .
```

And in Ensembl format:

```
 20   3   3   C/G   +
```

Individual 2

The second individual has the 3rd base deleted relative to the reference. In VCF, both the reference and variant allele columns must include the preceding base (T) and the reported position is that of the preceding base:

```
20   2   .   TC   T   .   PASS   .
```

In Ensembl format, the preceding base is not included, and the start/end coordinates represent the region of the sequence deleted. A "-" character is used to indicate that the base is deleted in the variant sequence:

```
20   3   3   C/-   +
```

The upshot of this is that while in the VCF input file the position of the variant is reported as 2, in the output file from VEP the position will be reported as 3. If no identifier is provided in the third column of the VCF, then the constructed identifier will be:

```
20_3_C/-
```

Individual 3

The third individual has an "A" inserted between the 3rd and 4th bases of the sequence relative to the reference. In VCF, as for the deletion, the base before the insertion is included in both the reference and variant allele columns, and the reported position is that of the preceding base:

```
20   3   .   C   CA   .   PASS   .
```

In Ensembl format, again the preceding base is not included, and the start/end positions are "swapped" to indicate that this is an insertion. Similarly to a deletion, a "-" is used to indicate no sequence in the reference:

```
 20   4   3   -/A   +
```

Again, the output will appear different, and the constructed identifier may not be what is expected:

```
20_3_-/A
```

Using VCF format output, or adding unique identifiers to the input (in the third VCF column), can mitigate this issue.

Complex VCF entries

For VCF entries with multiple alternate alleles, VEP will only trim the leading base from alleles if **all** REF and ALT alleles start with the same base:

```
20   3   .   C   CAAG,CAAGAAG   .   PASS   .
```

This will be considered internally by VEP as equivalent to:

```
20   4   3   -/AAG/AAGAAG   +
```

Now consider the case where a single VCF line contains a representation of both a SNV and an insertion:

```
20   3   .   C   CAAAG,G   .   PASS   .
```

Here the input alleles will remain unchanged, and VEP will consider the first REF/ALT pair as a substitution of C for CAAG, and the second as a C/G SNV:

```
20   3   3   C/CAAG/G   +
```

To modify this behaviour, VEP script users may use --minimal. This flag forces VEP to consider each REF/ALT pair independently, trimming identical leading and trailing bases from each as appropriate. Since this can lead to confusing output regarding coordinates etc, it is not the default behaviour. It is recommended to use the --allele_number flag to track the correspondence between alleles as input and how they appear in the output.

VCF - Structural variants

VEP can also call consequences on structural variants encoded in tab-delimited or VCF format. To recognise a variant as a structural variant, the allele string (or "SVTYPE" INFO field in VCF) must be set to one of the currently recognised values:

- **INS** - insertion
- **DEL** - deletion
- **DUP** - duplication
- **TDUP** - tandem duplication

Examples of structural variants encoded in tab-delimited format:

```
1    160283    471362    DUP   +  sv1
1    1385015   1387562   DEL   +  sv2
```

Examples of structural variants encoded in VCF format:

```
#CHROM  POS      ID   REF  ALT    QUAL  FILTER  INFO                      FORMAT
1       160283   sv1  .    <DUP>  .     .       SVTYPE=DUP;END=471362     .
1       1385015  sv2  .    <DEL>  .     .       SVTYPE=DEL;END=1387562    .
```

See the VCF definition document for more detail on how to describe structural variants in VCF format.

HGVS identifiers

See https://varnomen.hgvs.org for details. These must be relative to genomic or Ensembl transcript coordinates.

It also is possible to use RefSeq transcripts in both the web interface and the VEP script (see script documentation): this works for RefSeq transcripts that align to the genome correctly.

Examples:

```
ENST00000207771.3:c.344+626A>T
ENST00000471631.1:c.28_33delTCGCGG
ENST00000285667.3:c.1047_1048insC
5:g.140532T>C
```

Examples using RefSeq identifiers (using --refseq in the VEP script, or select the otherfeatures transcript database on the web interface and input type of HGVS):

```
NM_153681.2:c.7C>T
NM_005239.4:c.190G>A
NM_001025204.1:c.336G>A
```

HGVS protein notations may also be used, provided that they unambiguously map to a single genomic change. Due to redundancy in the amino acid code, it is not always possible to work out the corresponding genomic sequence change for a given protein sequence change. The following example is for a permissable protein notation in dog *(Canis familiaris)*:

```
ENSCAFP00000040171.1:p.Thr92Asn
```

HGVS notations may also be given in LRG coordinates:

```
LRG_1t1:c.841G>T
LRG_1:g.10006G>T
```

Variant identifiers

These should be e.g. dbSNP rsIDs, or any synonym for a variant present in the Ensembl Variation database. See here for a list of identifier sources in Ensembl.

Genomic SPDI notation

VEP can also support genomic SPDI notation which uses four fields delimited by colons S:P:D:I (Sequence:Position:Deletion:Insertion). See here for details.

Examples:

```
NC_000016.10:68684738:G:A
NC_000017.11:43092199:GCTTTT:
NC_000013.11:32315789::C
NC_000016.10:68644746:AA:GTA
16:68684738:2:AC
```

REST-style regions

VEP's region REST endoint requires variants are described as `[chr]:[start]-[end]:[strand]/[allele]`. This follows the same conventions as the default input format described above, with the key difference being that this format does not require the reference (REF) allele to be included; VEP will look up the reference allele using either a provided FASTA file (preferred) or Ensembl core database. Strand is optional and defaults to 1 (forward strand).

```
# SNP
5:140532-140532:1/C

# SNP (reverse strand)
14:19584687-19584687:-1/T

# insertion
1:881907-881906:1/C

# 5bp deletion
2:946507-946511:1/-
```

Output

VEP can return the results in different formats:

- Default VEP output
- Tab-delimited output
- VCF
- JSON output

Along with the results VEP computes and returns some statistics.

Default VEP output

The default output format ("VEP" format when downloading from the web interface) is a 14 column tab-delimited file. Empty values are denoted by '-'. The output columns are:

1. **Uploaded variation** - as chromosome_start_alleles
2. **Location** - in standard coordinate format (chr:start or chr:start-end)
3. **Allele** - the variant allele used to calculate the consequence
4. **Gene** - Ensembl stable ID of affected gene
5. **Feature** - Ensembl stable ID of feature
6. **Feature type** - type of feature. Currently one of Transcript, RegulatoryFeature, MotifFeature.
7. **Consequence** - consequence type of this variant
8. **Position in cDNA** - relative position of base pair in cDNA sequence
9. **Position in CDS** - relative position of base pair in coding sequence
10. **Position in protein** - relative position of amino acid in protein
11. **Amino acid change** - only given if the variant affects the protein-coding sequence
12. **Codon change** - the alternative codons with the variant base in upper case
13. **Co-located variation** - known identifier of existing variant
14. **Extra** - this column contains extra information as key=value pairs separated by ";", see below.

Other output fields:

- **REF_ALLELE** - the reference allele
- **IMPACT** - the impact modifier for the consequence type
- **VARIANT_CLASS** - Sequence Ontology variant class
- **SYMBOL** - the gene symbol
- **SYMBOL_SOURCE** - the source of the gene symbol
- **STRAND** - the DNA strand (1 or -1) on which the transcript/feature lies

- **ENSP** - the Ensembl protein identifier of the affected transcript
- **FLAGS** - transcript quality flags:
  - *cds_start_NF:* CDS 5' incomplete
  - *cds_end_NF:* CDS 3' incomplete
- **SWISSPROT** - Best match UniProtKB/Swiss-Prot accession of protein product
- **TREMBL** - Best match UniProtKB/TrEMBL accession of protein product
- **UNIPARC** - Best match UniParc accession of protein product
- **HGVSc** - the HGVS coding sequence name
- **HGVSp** - the HGVS protein sequence name
- **HGVSg** - the HGVS genomic sequence name
- **HGVS_OFFSET** - Indicates by how many bases the HGVS notations for this variant have been shifted
- **NEAREST** - Identifier(s) of nearest transcription start site
- **SIFT** - the SIFT prediction and/or score, with both given as prediction(score)
- **PolyPhen** - the PolyPhen prediction and/or score
- **MOTIF_NAME** - the source and identifier of a transcription factor binding profile aligned at this position
- **MOTIF_POS** - The relative position of the variation in the aligned TFBP
- **HIGH_INF_POS** - a flag indicating if the variant falls in a high information position of a transcription factor binding profile (TFBP)
- **MOTIF_SCORE_CHANGE** - The difference in motif score of the reference and variant sequences for the TFBP
- **CELL_TYPE** - List of cell types and classifications for regulatory feature
- **CANONICAL** - a flag indicating if the transcript is denoted as the canonical transcript for this gene
- **CCDS** - the CCDS identifer for this transcript, where applicable
- **INTRON** - the intron number (out of total number)
- **EXON** - the exon number (out of total number)
- **DOMAINS** - the source and identifer of any overlapping protein domains
- **DISTANCE** - Shortest distance from variant to transcript
- **IND** - individual name
- **ZYG** - zygosity of individual genotype at this locus
- **SV** - IDs of overlapping structural variants
- **FREQS** - Frequencies of overlapping variants used in filtering
- **AF** - Frequency of existing variant in 1000 Genomes
- **AFR_AF** - Frequency of existing variant in 1000 Genomes combined African population
- **AMR_AF** - Frequency of existing variant in 1000 Genomes combined American population
- **ASN_AF** - Frequency of existing variant in 1000 Genomes combined Asian population
- **EUR_AF** - Frequency of existing variant in 1000 Genomes combined European population
- **EAS_AF** - Frequency of existing variant in 1000 Genomes combined East Asian population
- **SAS_AF** - Frequency of existing variant in 1000 Genomes combined South Asian population
- **AA_AF** - Frequency of existing variant in NHLBI-ESP African American population
- **EA_AF** - Frequency of existing variant in NHLBI-ESP European American population
- **gnomAD_AF** - Frequency of existing variant in gnomAD exomes combined population
- **gnomAD_AFR_AF** - Frequency of existing variant in gnomAD exomes African/American population
- **gnomAD_AMR_AF** - Frequency of existing variant in gnomAD exomes American population
- **gnomAD_ASJ_AF** - Frequency of existing variant in gnomAD exomes Ashkenazi Jewish population
- **gnomAD_EAS_AF** - Frequency of existing variant in gnomAD exomes East Asian population
- **gnomAD_FIN_AF** - Frequency of existing variant in gnomAD exomes Finnish population
- **gnomAD_NFE_AF** - Frequency of existing variant in gnomAD exomes Non-Finnish European population
- **gnomAD_OTH_AF** - Frequency of existing variant in gnomAD exomes combined other combined populations

- **gnomAD_SAS_AF** - Frequency of existing variant in gnomAD exomes South Asian population
- **MAX_AF** - Maximum observed allele frequency in 1000 Genomes, ESP and gnomAD
- **MAX_AF_POPS** - Populations in which maximum allele frequency was observed
- **CLIN_SIG** - ClinVar clinical significance of the dbSNP variant
- **BIOTYPE** - Biotype of transcript or regulatory feature
- **APPRIS** - Annotates alternatively spliced transcripts as primary or alternate based on a range of computational methods. NB: not available for GRCh37
- **TSL** - Transcript support level. NB: not available for GRCh37
- **PUBMED** - Pubmed ID(s) of publications that cite existing variant
- **SOMATIC** - Somatic status of existing variant(s); multiple values correspond to multiple values in the Existing_variation field
- **PHENO** - Indicates if existing variant is associated with a phenotype, disease or trait; multiple values correspond to multiple values in the Existing_variation field
- **GENE_PHENO** - Indicates if overlapped gene is associated with a phenotype, disease or trait
- **ALLELE_NUM** - Allele number from input; 0 is reference, 1 is first alternate etc
- **MINIMISED** - Alleles in this variant have been converted to minimal representation before consequence calculation
- **PICK** - indicates if this block of consequence data was picked by --flag_pick or --flag_pick_allele
- **BAM_EDIT** - Indicates success or failure of edit using BAM file
- **GIVEN_REF** - Reference allele from input
- **USED_REF** - Reference allele as used to get consequences
- **REFSEQ_MATCH** - the RefSeq transcript match status; contains a number of flags indicating whether this RefSeq transcript matches the underlying reference sequence and/or an Ensembl transcript (more information).
  - *rseq_3p_mismatch:* signifies a mismatch between the RefSeq transcript and the underlying primary genome assembly sequence. Specifically, there is a mismatch in the 3' UTR of the RefSeq model with respect to the primary genome assembly (e.g. GRCh37/GRCh38).
  - *rseq_5p_mismatch:* signifies a mismatch between the RefSeq transcript and the underlying primary genome assembly sequence. Specifically, there is a mismatch in the 5' UTR of the RefSeq model with respect to the primary genome assembly.
  - *rseq_cds_mismatch:* signifies a mismatch between the RefSeq transcript and the underlying primary genome assembly sequence. Specifically, there is a mismatch in the CDS of the RefSeq model with respect to the primary genome assembly.
  - *rseq_ens_match_cds:* signifies that for the RefSeq transcript there is an overlapping Ensembl model that is identical across the CDS region only. A CDS match is defined as follows: the CDS and peptide sequences are identical and the genomic coordinates of every translatable exon match. Useful related attributes are: rseq_ens_match_wt and rseq_ens_no_match.
  - *rseq_ens_match_wt:* signifies that for the RefSeq transcript there is an overlapping Ensembl model that is identical across the whole transcript. A whole transcript match is defined as follows: 1) In the case that both models are coding, the transcript, CDS and peptide sequences are all identical and the genomic coordinates of every exon match. 2) In the case that both transcripts are non-coding the transcript sequences and the genomic coordinates of every exon are identical. No comparison is made between a coding and a non-coding transcript. Useful related attributes are: rseq_ens_match_cds and rseq_ens_no_match.
  - *rseq_ens_no_match:* signifies that for the RefSeq transcript there is no overlapping Ensembl model that is identical across either the whole transcript or the CDS. This is caused by differences between the transcript, CDS or peptide sequences or between the exon genomic coordinates. Useful related attributes are: rseq_ens_match_wt and rseq_ens_match_cds.
  - *rseq_mrna_match:* signifies an exact match between the RefSeq transcript and the underlying primary genome assembly sequence (based on a match between the transcript stable id and an accession in the RefSeq mRNA file). An exact match occurs when the underlying genomic sequence of the model can be perfectly aligned to the mRNA sequence post polyA clipping.
  - *rseq_mrna_nonmatch:* signifies a non-match between the RefSeq transcript and the underlying primary genome assembly sequence. A non-match is deemed to have occurred if the underlying genomic sequence does not have a perfect alignment to the mRNA sequence post polyA clipping. It can also signify that no comparison was possible as the model stable id may not have had a corresponding entry in the RefSeq mRNA file (sometimes happens when accessions are retired or changed). When a non-match occurs one or several of the following transcript attributes will also be present to provide more detail on the nature of the non-match: rseq_5p_mismatch, rseq_cds_mismatch, rseq_3p_mismatch, rseq_nctran_mismatch, rseq_no_comparison
  - *rseq_nctran_mismatch:* signifies a mismatch between the RefSeq transcript and the underlying primary genome assembly sequence. This is a comparison between the entire underlying genomic sequence of the RefSeq model to the mRNA in the case of RefSeq models that are non-coding.

- *rseq_no_comparison:* signifies that no alignment was carried out between the underlying primary genome assembly sequence and a corresponding RefSeq mRNA. The reason for this is generally that no corresponding, unversioned accession was found in the RefSeq mRNA file for the transcript stable id. This sometimes happens when accessions are retired or replaced. A second possibility is that the sequences were too long and problematic to align (though this is rare).

- **OverlapBP** - Number of base pairs overlapping with the corresponding structural variation feature
- **OverlapPC** - Percentage of corresponding structural variation feature overlapped by the given input
- **CHECK_REF** - Reports variants where the input reference does not match the expected reference
- **AMBIGUITY** - IUPAC allele ambiguity code

Example of VEP default output format:

```
11_224088_C/A    11:224088   A  ENSG00000142082  ENST00000525319  Transcript         missense_va
11_224088_C/A    11:224088   A  ENSG00000142082  ENST00000534381  Transcript         5_prime_UTR
11_224088_C/A    11:224088   A  ENSG00000142082  ENST00000529055  Transcript         downstream_
11_224585_G/A    11:224585   A  ENSG00000142082  ENST00000529937  Transcript         intron_vari
22_16084370_G/A  22:16084370 A  -                ENSR00000615113  RegulatoryFeature  regulatory_
```

The VEP script will also add a header to the output file. This contains information about the databases connected to, and also a key describing the key/value pairs used in the extra column.

```
## ENSEMBL VARIANT EFFECT PREDICTOR v104.0
## Output produced at 2017-03-21 14:51:27
## Connected to homo_sapiens_core_104_38 on ensembldb.ensembl.org
## Using cache in /homes/user/.vep/homo_sapiens/104_GRCh38
## Using API version 104, DB version 104
## polyphen version 2.2.2
## sift version sift5.2.2
## COSMIC version 78
## ESP version 20141103
## gencode version GENCODE 25
## genebuild version 2014-07
## HGMD-PUBLIC version 20162
## regbuild version 16
## assembly version GRCh38.p7
## ClinVar version 201610
## dbSNP version 147
## Column descriptions:
## Uploaded_variation : Identifier of uploaded variant
## Location : Location of variant in standard coordinate format (chr:start or chr:start-end)
## Allele : The variant allele used to calculate the consequence
## Gene : Stable ID of affected gene
## Feature : Stable ID of feature
## Feature_type : Type of feature - Transcript, RegulatoryFeature or MotifFeature
## Consequence : Consequence type
## cDNA_position : Relative position of base pair in cDNA sequence
## CDS_position : Relative position of base pair in coding sequence
## Protein_position : Relative position of amino acid in protein
## Amino_acids : Reference and variant amino acids
## Codons : Reference and variant codon sequence
## Existing_variation : Identifier(s) of co-located known variants
## Extra column keys:
## IMPACT : Subjective impact classification of consequence type
## DISTANCE : Shortest distance from variant to transcript
## STRAND : Strand of the feature (1/-1)
## FLAGS : Transcript quality flags
```

Tab-delimited output

The --tab flag instructs VEP to write output as a tab-delimited table.
This differs from the default output format in that each individual field from the "Extra" field is written to a separate tab-delimited column.
This makes the output more suitable for import into spreadsheet programs such as Excel.

Furthermore the header is the same as the one for the VEP default output format and this is also the format used when selecting the "TXT" option on the VEP web interface.

Example of VEP tab-delimited output format:

```
#Uploaded_variation  Location   Allele  Gene             Feature          Feature_type  Conseque
11_224088_C/A        11:224088  A       ENSG00000142082  ENST00000525319  Transcript    missense
11_224088_C/A        11:224088  A       ENSG00000142082  ENST00000534381  Transcript    downstre
11_224088_C/A        11:224088  A       ENSG00000142082  ENST00000529055  Transcript    downstre
11_224585_G/A        11:224585  A       ENSG00000142082  ENST00000529937  Transcript    intron_
```

The choice and order of columns in the output may be configured using --fields. For instance:

```
./vep -i examples/homo_sapiens_GRCh38.vcf --cache --force_overwrite --tab --fields "Uploaded_va
```

VCF output

The VEP script can also generate VCF output using the --vcf flag.

Main information about the specificity of the VEP VCF output format:

- Consequences are added in the INFO field of the VCF file, using the key "**CSQ**" (you can change it using --vcf_info_field).
- Data fields are encoded separated by the character "**|**" (pipe). The order of fields is written in the VCF header. Unpopulated fields are represented by an empty string.
- Output fields in the "CSQ" INFO field can be configured by using --fields.
- Each prediction, for a given variant, is separated by the character "**,**" in the CSQ INFO field (e.g. when a variant overlaps more than 1 transcript)

Here is a list of the (default) fields you can find within the CSQ field:

```
Allele|Consequence|IMPACT|SYMBOL|Gene|Feature_type|Feature|BIOTYPE|EXON|INTRON|HGVSc|HGVSp|cDNA_
```

Example of VEP command using the --vcf and --fields options:

```
./vep -i examples/homo_sapiens_GRCh38.vcf --cache --force_overwrite --vcf --fields "Allele,Conse
```

VCFs produced by VEP can be filtered by filter_vep.pl in the same way as standard format output files.

If the input format was VCF, the file will remain unchanged save for the addition of the CSQ field and the header (unless using any filtering). If an existing CSQ field is found, it will be replaced by the one added by the VEP (use --keep_csq to preserve it).

Custom data added with --custom are added as separate fields, using the key specified for each data file.

Commas in fields are replaced with ampersands (&) to preserve VCF format.

```
##INFO=<ID=CSQ,Number=.,Type=String,Description="Consequence annotations from Ensembl VEP. Forma
#CHROM  POS       ID          REF  ALT  QUAL  FILTER  INFO
21      26978790  rs75377686  T    C    .     .       CSQ=C|missense_variant|MODERATE|MRPL39|ENS
```

JSON output

VEP can produce output in the form of serialised JSON objects using the --json flag. JSON is a serialisation format that can be parsed and processed easily by many packages and programming languages; it is used as the default output format for Ensembl's REST server.

Each input variant is reported as a single JSON object which constitutes one line of the output file. The JSON object is structured somewhat differently to the other VEP output formats, in that per-variant fields (e.g. co-located existing variant details) are reported only once. Consequences are grouped under the feature type that they affect (Transcript, Regulatory Feature, etc). The original

input line (e.g. from VCF input) is reported under the "input" key in order to aid aligning input with output. When using a cache file, frequencies for co-located variants are reported by default (see --af_1kg, --af_esp, --af_gnomad, --af_exac).

Here follows an example of JSON output (prettified and redacted for display here):


```
{
  "input": "1 1918090 test1 A G . . .",
  "id": "test1",
  "seq_region_name": "1",
  "start": 1918090,
  "end": 1918090,
  "strand": 1,
  "allele_string": "A/G",
  "most_severe_consequence": "missense_variant",
  "colocated_variants": [
    {
      "id": "COSV57068665",
      "seq_region_name": "1",
      "start": 1918090,
      "end": 1918090,
      "strand": 1,
      "allele_string": "COSMIC_MUTATION"
    },
    {
      "id": "rs28640257",
      "seq_region_name": "1",
      "start": 1918090,
      "end": 1918090,
      "strand": 1,
      "allele_string": "A/G/T",
      "minor_allele": "G",
      "minor_allele_freq": 0.352,
      "frequencies": {
        "G": {
          "amr": 0.5072,
          "gnomad_sas": 0.369,
          "gnomad": 0.4541,
          "ea": 0.4986,
          "gnomad_oth": 0.4611,
          "gnomad_asj": 0.3909,
          "gnomad_nfe": 0.4944,
          "aa": 0.1207,
          "gnomad_afr": 0.103,
          "afr": 0.053,
          "gnomad_amr": 0.5641,
          "gnomad_fin": 0.474,
          "sas": 0.3906,
          "gnomad_eas": 0.4598,
          "eur": 0.4901,
          "eas": 0.4623
        }
      }
    }
  ],
  "transcript_consequences": [
    {
      "variant_allele": "G",
      "consequence_terms": [
        "missense_variant"
      ],
      "gene_id": "ENSG00000178821",
      "transcript_id": "ENST00000310991",
      "strand": -1,
      "cdna_start": 436,
      "cdna_end": 436,
      "cds_start": 422,
      "cds_end": 422,
      "protein_start": 141,
      "protein_end": 141,
      "codons": "aTg/aCg",
```

```
      "amino_acids": "M/T",
      "polyphen_prediction": "benign",
      "polyphen_score": 0.001,
      "sift_prediction": "tolerated",
      "sift_score": 0.22,
      "hgvsp": "ENSP00000311122.3:p.Met141Thr",
      "hgvsc": "ENST00000310991.8:c.422T>C"
      }
  ],
  "regulatory_feature_consequences": [
    {
      "variant_allele": "G",
      "consequence_terms": [
        "regulatory_region_variant"
      ],
      "regulatory_feature_id": "ENSR00000000255"
    }
  ]
}
```


In accordance with JSON conventions, all keys (except alleles) are lower-case. Some keys also have different names and structures to those found in the other VEP output formats:

Key	JSON equivalent(s)	Notes
Consequence	consequence_terms	
Gene	gene_id	
Feature	transcript_id, regulatory_feature_id, motif_feature_id	Consequences are grouped under the feature type they affect
ALLELE	variant_allele	
SYMBOL	gene_symbol	
SYMBOL_SOURCE	gene_symbol_source	
ENSP	protein_id	
OverlapBP	bp_overlap	
OverlapPC	percentage_overlap	
Uploaded_variation	id	
Location	seq_region_name, start, end, strand	The variant's location field is broken down into constituent coordinate parts for clarity. "seq_region_name" is used in place of "chr" or "chromosome" for consistency with other parts of Ensembl's REST API
*_maf	*_allele, *_maf	
cDNA_position	cdna_start, cdna_end	
CDS_position	cds_start, cds_end	
Protein_position	protein_start, protein_end	
SIFT	sift_prediction, sift_score	
PolyPhen	polyphen_prediction, polyphen_score	

Statistics

VEP writes an HTML file containing statistics pertaining to the results of your job; it is named **[output_file]_summary.html** (with the default options the file will be named **variant_effect_output.txt_summary.html**). To view it you should open the file in your web browser.

To prevent VEP writing a stats file, use the flag --no_stats. To have VEP write a machine-readable text file in place of the HTML, use --stats_text. To change the name of the stats file from the default, use --stats_file [file].

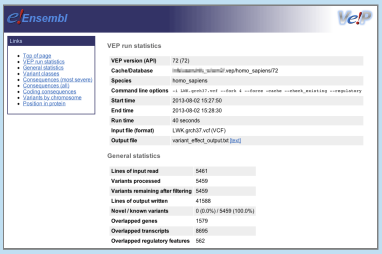
The page contains several sections:

General statistics

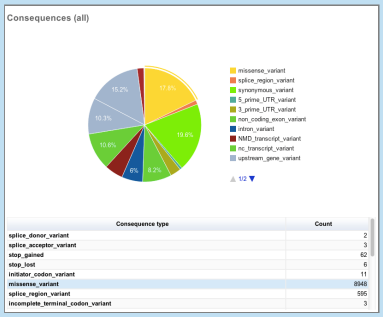
This section contains two tables. The first describes the cache and/or database used, the version of VEP, species, command line parameters, input/output files and run time. The second table contains information about the number of variants, and the number of genes, transcripts and regulatory features overlapped by the input.

Charts and tables

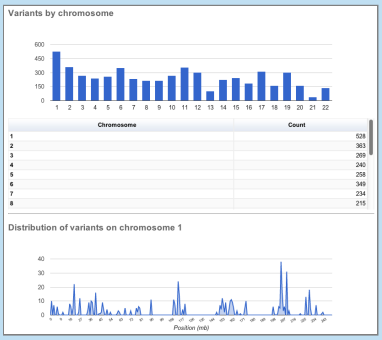
There then follows several charts, most with accompanying tables. Tables and charts are interactive; clicking on a row to highlight it in the table will highlight the relevant segment in the chart, and vice versa.



General statistics



Summary of called consequence types



Distribution of variants across chromosomes

Variant Effect Predictor FAQ

For any questions not covered here, please send an email to the Ensembl developer's mailing list (public) or contact the Ensembl Helpdesk (private).

General questions

Q: Why has my insertion/deletion variant encoded in VCF disappeared from the VEP output?

Ensembl treats unbalanced variants differently to VCF - your variant hasn't disappeared, it may have just changed slightly! You can solve this by giving your variants a unique identifier in the third column of the VCF file. See here for a full discussion.

Q: Why don't I see any co-located variants when using species X?

Ensembl only has variation databases for a subset of all Ensembl species - see this document for details.

Q: Why do I see multiple known variants mapped to my input variant?

VEP compares you input to known variants from the Ensembl variation database. In some cases one input variant can match multiple known variants:

- Germline variants from dbSNP and somatic mutations from COSMIC may be found at the same locus
- Some sources, e.g. HGMD, do not provide public access to allele-specific data, so an HGMD variant with unknown alleles may colocate with one from dbSNP with known alleles
- Multiple alternate alleles from your input may match different variants as they are described in dbSNP

See here for a full discussion.

Q: VEP is not assigning a frequency to my input variant - why?

VEP's cache contains frequency data only for variants and alleles imported into Ensembl's variation database. See here for a full discussion.

Q: Why do I see so many lines of output for each variant in my input?

While it would be convenient to have a simple, one word answer to the question "What is the consequence of this variant?", in reality biology is not this simple! Many genes have more than one transcript, so VEP provides a prediction for each transcript that a variant overlaps. VEP has options to help select results according to your requirements; the --canonical and --ccds options indicate which transcripts are canonical and belong to the CCDS set respectively, while --pick, --per_gene, --summary and --most_severe allow you to give a more summary level assessment per variant.

Furthermore, several "compound" consequences are also possible - if, for example, a variant falls in the final few bases of an exon, it may be considered to affect a splicing site, in addition to possibly affecting the coding sequence.

Q: How do I reduce VEP's memory requirement?

There are a number of ways to do this-

1. Ensure your input file is sorted by location. This can greatly reduce memory requirements and runtime
2. Consider reducing the buffer size. This reduces the number of variants annotated together in a batch and can be modified in both command line and web interfaces. Reducing buffer size may increase run time.
3. Ensure you are only using the options you need, rather than --everything. Some data-rich options, such as regulatory annotation have an impact on memory use

Web VEP questions

Q: How do I access the web version of the Variant Effect Predictor?

You can find the web VEP on the Tools page.

Q: Why is the output I get for my input file different when I use the web VEP and command line VEP?

Ensure that you are passing equivalent arguments to the script that you are using in the web version. If you are sure this is still a problem, please report it on the ensembl-dev mailing list.

Command line VEP questions

Q: How can I make VEP run faster?

There are a number of factors that influence how fast VEP runs. Have a look at our handy guide for tips on improving VEP runtime.

Q: Why do I see "N" as the reference allele in my HGVS strings?

Q: Why do I see the following error (or similar) in my VEP output?

```
substr outside of string at /nfs/users/nfs_w/wm2/Perl/ensembl-variation/modules/Bio/EnsEMBL/Vari
Use of uninitialized value $ref_allele in string eq at /nfs/users/nfs_w/wm2/Perl/ensembl-variati
Use of uninitialized value in concatenation (.) or string at /nfs/users/nfs_w/wm2/Perl/ensembl-v
```

Both of these error types are usually seen when using a FASTA file for retrieving sequence. There are a couple of steps you can take to try to remedy them:

1. The index alongside the FASTA can become corrupted. Delete [fastafile].index and re-run VEP to regenerate it. By default this file is located in your $HOME/.vep/[species]/[version]_[assembly] directory.
2. The FASTA file itself may have been corrupted during download; delete the fasta file and the index and re-download (you can use the VEP installer to do this).
3. Older versions of BioPerl (1.2.3 in particular is known to have this) cannot properly index large FASTA files. Make sure you are using a later (>=1.6) version of BioPerl. The VEP installer installs 1.6.924 for you.

If you still see problems after taking these steps, or if you were not using a FASTA file in the first place, please contact us.

Q: Why do I see the following warning?

```
WARNING: Chromosome 21 not found in annotation sources or synonyms on line 160
```

This can occur if the chromosome names differ between your input variant and any annotation source that you are using (cache, database, GFF/GTF file, FASTA file, custom annotation file). To circumvent this you may provide VEP with a synonyms file. A synonym file is included in VEP's cache files, so if you have one of these for your species you can use it as follows:

```
./vep -i input.vcf -cache -synonyms ~/.vep/homo_sapiens/104_GRCh38/chr_synonyms.txt
```

The file consists of lines containing pairs of tab-separated synonyms. Order is not important as synonyms can be used in both "directions".

Q: Can I get gnomAD or ExAC allele frequencies in VEP?

Yes, see this guide.

Q: Why do I see the following error?

```
Could not connect to database homo_sapiens_core_63_37 as user anonymous using [DBI:mysql:databas
Unknown MySQL server host 'ensembldb.ensembl.org' (2) at $HOME/src/ensembl/modules/Bio/EnsEMBL/D

------------------- EXCEPTION --------------------
MSG: Could not connect to database homo_sapiens_core_63_37 as user anonymous using [DBI:mysql:da
Unknown MySQL server host 'ensembldb.ensembl.org' (2)
```

By default VEP is configured to connect to the public MySQL server at ensembldb.ensembl.org. Occasionally the server may break connection with your process, which causes this error. This can happen when the server is busy, or due to various network issues. Consider using a local copy of the database, or the caching system.

Q: Can I use VEP on Windows?

Yes - see the documentation for a few different ways to get the VEP running on Windows.

Q: Can I download all of the SIFT and/or PolyPhen predictions?

The Ensembl Variation database and the human VEP cache file contain precalculated SIFT and PolyPhen-2 predictions for every possible amino acid change in every translated protein product in Ensembl. Since these data are huge, we store them in a compressed format. The best approach to extract them is to use our Perl API.

The format in which the data are stored in our database is described here

The simplest way to access these matrices is to use an API script to fetch a ProteinFunctionPredictionMatrix for your protein of interest and then call its 'get_prediction' method to get the score for a particular position and amino acid, looping over all possible amino acids for your position. There is some detailed documentation on this class in the API documentation here.

You would need to work out which peptide position your codon maps to, but there are methods in the TranscriptVariationAllele class that should help you (probably translation_start and translation_end).